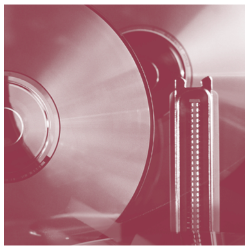
ASG-Manager Products™
User Defined Syntax

Version: 2.5

Publication Number: MPR2100-25-UDS

Publication Date: December 2000



ASG Worldwide Headquarters Naples, Florida USA | asg.com

1333 Third Avenue South, Naples, Florida 34102 USA Tel: 941.435.2200 Fax: 941.263.3692 Toll Free: 1.800.932.5536

ASG Documentation/Product Enhancement Fax Form

Please FAX comments regarding ASG products and/or documentation to (941) 263-3692.

Company Name	Telephone Number	Site ID	Contact name

Product Name/Publication	Version #	Publication Date
Product:		
Publication:		
Tape VOLSER:		

Enhancement Request:

ASG Support Numbers

ASG provides support throughout the world to resolve questions or problems regarding installation, operation, or use of our products. We provide all levels of support during normal business hours and emergency support during non-business hours. To expedite response time, please follow these procedures.

Please have this information ready:

- Product name, version number, and release number
- List of any fixes currently applied
- Any alphanumeric error codes or messages written precisely or displayed
- A description of the specific steps that immediately preceded the problem
- The severity code (ASG Support uses an escalated severity system to prioritize service to our clients. The severity codes and their meanings are listed below.)

If You Receive a Voice Mail Message:

1. Follow the instructions to report a production-down or critical problem.
2. Leave a detailed message including your name and phone number. A Support representative will be paged and will return your call as soon as possible.
3. Please have the information described above ready for when you are contacted by the Support representative.

Severity Codes and Expected Support Response Times

Severity	Meaning	Expected Support Response Time
1	Production down, critical situation	Within 30 minutes
2	Major component of product disabled	Within 2 hours
3	Problem with the product, but customer has work-around solution	Within 4 hours
4	"How-to" questions and enhancement requests	Within 4 hours

ASG provides software products that run in a number of third-party vendor environments. Support for all non-ASG products is the responsibility of the respective vendor. In the event a vendor discontinues support for a hardware and/or software product, ASG cannot be held responsible for problems arising from the use of that unsupported version.

Business Hours Support

Your Location	Phone	Fax	E-mail
United States and Canada	800.354.3578 1.941.435.2201 **Secondary Numbers:** 800.227.7774 800.525.7775	941.263.2883	support@asg.com
Australia	61.2.9460.0411	61.2.9460.0280	support.au@asg.com
England	44.1727.736305	44.1727.812018	support.uk@asg.com
France	33.141.028590	33.141.028589	support.fr@asg.com
Germany	49.89.45716.300	49.89.45716.400	support.de@asg.com
Singapore	65.224.3080	65.224.8516	support.sg@asg.com
All other countries:	1.941.435.2201		support@asg.com

Non-Business Hours - Emergency Support

Your Location	Phone	Your Location	Phone
United States and Canada	800.354.3578 1.941.435.2201 **Secondary Numbers:** 800.227.7774 800.525.7775 **Fax:** 941.263.2883		
Asia	011.65.224.3080	**Japan/Telecom**	0041.800.9932.5536
Australia	0011.800.9932.5536	**New Zealand**	00.800.9932.5536
Denmark	00.800.9932.5536	**South Korea**	001.800.9932.5536
France	00.800.9932.5536	**Sweden/Telia**	009.800.9932.5536
Germany	00.800.9932.5536	**Switzerland**	00.800.9932.5536
Hong Kong	001.800.9932.5536	**Thailand**	001.800.9932.5536
Ireland	00.800.9932.5536	**United Kingdom**	00.800.9932.5536
Israel/Bezeq	014.800.9932.5536		
Japan/IDC	0061.800.9932.5536	**All other countries**	1.941.435.2201

ASG Web Site

Visit http://www.asg.com, ASG's World Wide Web site.

Submit all product and documentation suggestions to ASG's product management team at http://www.asg.com/products/suggestions.asp

If you do not have access to the web, FAX your suggestions to product management at (941) 263-3692. Please include your name, company, work phone, e-mail ID, and the name of the ASG product you are using. For documentation suggestions include the publication number located on the publication's front cover.

Contents

Preface

This *ASG-Manager Products User Defined Syntax* publication describes the optional User Defined Syntax (herein called UDS) facility (selectable unit CMR-UD1) whereby dictionary member types and their attributes can be defined and set up in user installations to the installation's own requirements.

Allen Systems Group, Inc. (ASG) provides professional support to resolve any questions or concerns regarding the installation or use of any ASG product. Telephone technical support is available around the world, 24 hours a day, 7 days a week.

ASG welcomes your comments, as a preferred or prospective customer, on this publication or on any ASG product.

About this Publication

This publication consists of these chapters:

- Chapter 1, "Introduction to UDS," describes UDS and its different hierarchies.
- Chapter 2, "Defining a Hierarchy Member," describes the different clauses and member-types in the hierarchy member.
- Chapter 3, "Defining a Member-type Member," describes the different clauses and member-types in a member-type member.
- Chapter 4, "Defining a Member-type-group Member," describes the member-type-group member.
- Chapter 5, "Defining an Attribute Type," provides details on the attribute types used in UDS tables.
- Chapter 6, "Defining an Attribute Group," describes how to define an attribute group.
- Chapter 7, "Interrogating the Contents of a UDS Table," describes how to interrogate the contents of a UDS table through the use of the SHOW UDS command.

- Chapter 8, "Looking Ahead: Planning for Future Attributes," describes how to plan for the addition of attributes and new UDS tables.
- Chapter 9, "Common Clauses," provides reference syntax for commonly used clauses.
- Chapter 10, "UDS Load Modules Supplied by ASG," provides a listing of the UDS load modules supplied by ASG and the ASG-Manager Products that they support.

Publication Conventions

These conventions apply to syntax diagrams that appear in this publication.

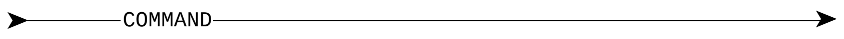
Diagrams are read from left to right along a continuous line (the "main path"). Keywords and variables appear on, above, or below the main path.

Convention	Represents
➤➤	at the beginning of a line indicates the start of a statement.
➤◄	at the end of a line indicates the end of a statement.
———➤	at the end of a line indicates that the statement continues on the line below.
➤———	at the beginning of a line indicates that the statement continues from the line above.

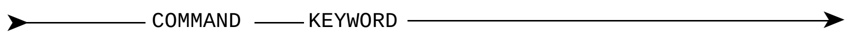
Keywords are in upper-case characters. Keywords and any required punctuation characters or symbols are highlighted. Permitted truncations are not indicated.

Variables are in lower-case characters.

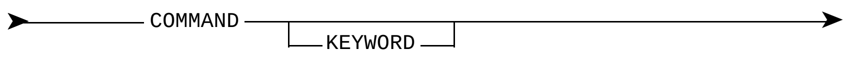
Statement identifiers appear on the main path of the diagram:



A required keyword appears on the main path:

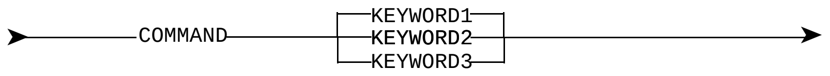


An optional keyword appears below the main path:

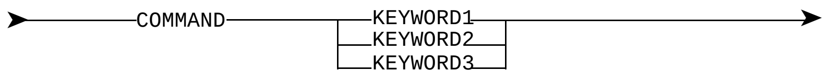


Where there is a choice of required keywords, the keywords appear in a vertical list; one of them is on the main path:

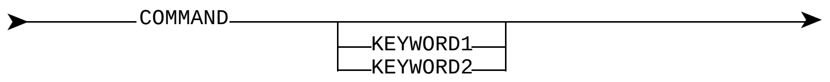
Convention	Represents



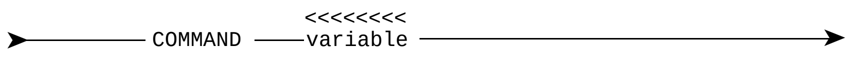
or



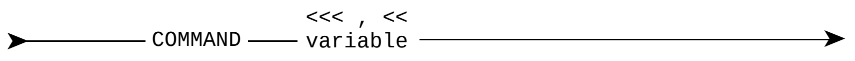
Where there is a choice of optional keywords, the keywords appear in a vertical list, below the main path:



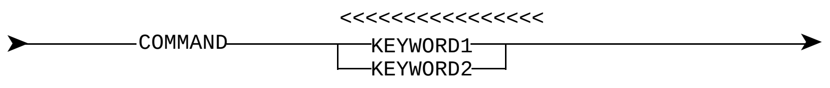
The repeat symbol, <<<<<<, above a keyword or variable, or above a whole clause, indicates that the keyword, variable, or clause may be specified more than once:



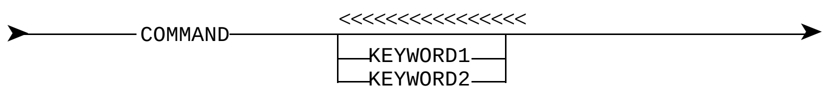
A repeat symbol broken by a comma indicates that if the keyword, variable, or clause is specified more than once, a comma must separate each instance of the keyword, variable, or clause:



The repeat symbol above a list of keywords (one of which appears on the main path) indicates that any one or more of the keywords may be specified; at least one must be specified:



The repeat symbol above a list of keywords (all of which are below the main path) indicates that any one or more of the keywords maybe specified, but they are all optional:



Allen Systems Group, Inc. uses these conventions in publications:

Convention	**Represents**
ALL CAPITALS	Directory, path, file, dataset, member, database, program, command, and parameter names.
Initial Capitals on Each Word	Window, field, field group, check box, button, panel (or screen), option names, and names of keys. A plus sign (+) is inserted for key combinations (e.g., Alt+Tab).
`lowercase italic monospace`	Information that you provide according to your particular situation. For example, you would replace *`filename`* with the actual name of the file.
`Monospace`	Characters you must type exactly as they are shown. Code, JCL, file listings, or command/statement syntax. Also used for denoting brief examples in a paragraph.
Vertical Separator Bar (|) with underline	Options available with the default value underlined (e.g., Y|<u>N</u>).

1 Introduction to UDS

This chapter includes these sections:

How UDS Can Help You

The UDS facility lets you define your own member types and attributes for use in your dictionary. Using the facility you can tailor the dictionary so that it precisely matches the specific requirements of your organization. For example:

- You can rename member types, or create new member types, to reflect the terminology in use in your organization. This feature is particularly valuable not only in non-English speaking countries but also at sites whose terminology does not match that in use in the standard ASG-Manager Products software.
- You can create entirely new member types, outside the standard range of member types provided by ASG, to support your own dictionary requirements and computer applications.
- You can define the relationships that can be supported between member types, within the general structure of relationships permitted by ASG.
- You can create new attributes for inclusion in the members of the dictionary, specifying your own validation rules for the values of such attributes if you wish.

Member Types Supplied by ASG

The ControlManager Nucleus enables you to encode any member types supplied by ASG, except the DataManager Database Management System Interfaces and MARK IV Interface member types. For these DataManager member types to be available to you, you will need to have the relevant selectable units installed. Please turn to Appendix A, "Relationships Permitted from Base Member Types," on page 67 to see a list of these member types.

Users with UDS can define their own member types. In addition, ASG supplies to such users three preferred sets of member types. These are extensions of the basic set of member types used by DataManager (SYSTEM, PROGRAM, MODULE, FILE, GROUP, ITEM), containing member types commonly used in the data processing environments. Non-English language versions of one of these preferred sets of member types, the EDPS set, are also available. These extended sets of member types can be installed without the need to create your own member types. See "Extensions of the Basic Set of Member Types" on page 4 through "The SDS Hierarchy" on page 7.

Creating a UDS Table

The structure of user defined member types available in a given dictionary is defined in a table held on the MP-AID known as a *UDS table*. The UDS table must be constructed onto the MP-AID by the dictionary Controller or by an individual specifically authorized to perform that function by the Controller. An overview of the procedure for constructing a UDS table to the MP-AID is given in the *ASG-Manager Products Systems Administrator's Manual*, and the requisite commands are fully documented in the *ASG-Manager Products Controller's Manual*.

A UDS table is constructed from these members created in the ASG-Manager Products Administration Dictionary:

- A HIERARCHY member which specifies the user defined member types that are to be contained in the UDS table.
- One or more MEMBER-TYPE members that define the member types that are to be available in the UDS table. Each user defined member type must be modeled on one of the member types supplied by ASG. These ASG supplied member types are called *base member types*.
- Optionally, one or more MEMBER-TYPE-GROUP members that group member types together so that they can be collectively referred to.
- Optionally, one or more ATTRIBUTE-TYPE members to define new attributes for incorporation in the member types in the UDS table.
- Optionally, one or more ATTRIBUTE-GROUP members that group attributes together so that they can be collectively referred to.

These are all dictionary members and can be created by any user authorized to update the ASG-Manager Products Administration Dictionary.

Creation and maintenance of the ASG-Manager Products Administration Dictionary are discussed in the *ASG-Manager Products Systems Administrator's Manual*, and the commands for creating the dictionary and restoring ASG-supplied members into it are given in *ASG-Manager Products Controller's Manual*.

The ASG-Manager Products release tape includes a SAVE file of an ASG-supplied dictionary, the *UDS Table Dictionary* . This dictionary contains the data definitions of members representing standard UDS tables for different Manager Products configurations (including the UDS table supplied for use with the ASG-Manager Products Administration Dictionary).

You are advised to use this UDS Table Dictionary as a basis for any further UDS table development.

The various UDS tables represented by the members in the UDS Table Dictionary are also supplied as load modules.

See Chapter 10, "UDS Load Modules Supplied by ASG," on page 65 for a list of the load modules supplied.

Interrogating the UDS Table

The UDS table for the current dictionary can be interrogated using the SHOW UDS command. The command can output all, or any part of, this information:

- The keywords that identify each MEMBER-TYPE in the dictionary
- The literals specified for each MEMBER-TYPE
- The hierarchical positions of the MEMBER-TYPEs
- The relationships allowed between MEMBER-TYPEs
- The attributes relating to each MEMBER-TYPE
- The allowed values of the attributes and other details
- The format line numbers and parameters by which the elements in the UDS table can be referred to in a user defined report (if the User Defined Output facility is installed).

The SHOW UDS command is fully documented in Chapter 7, "Interrogating the Contents of a UDS Table," on page 59.

Extensions of the Basic Set of Member Types

There are three ASG-supplied extensions of the basic set of member types:

- EDPS (Extended Data Processing Structure)
- SAS (Structured Analysis Structure)
- SDS (Structured Development Structure)

To see a diagram of these structures, turn to "The EDPS Hierarchy" on page 5, "The SAS Hierarchy" on page 7, and "The SDS Hierarchy" on page 7, respectively. In the diagrams of the structures, a number that represents its level number appears against each member type. The top member type in a group has a level number of 10, the next member type down a level number of 20 and so on, in increments of 10 until the top member type in the next group is encountered. The system of level numbers restricts the members to which a member of any given type may refer.

If a member refers to a member that does not already exist in the data entries data set, then a dummy member is created. The type of dummy created is usually the same as the member that would have been created by a reference from the same clause in a member of the top level member type in the group. However occasionally, the type of dummy may not be the same. For example, a dummy created from a reference in a SYSTEM CONTAINS clause is a PROGRAM and not an ITEM.

If the GENERIC keyword is included in a command that selects members by member type, the GENERIC keyword can be followed by the name of any of the top level member types in a group.

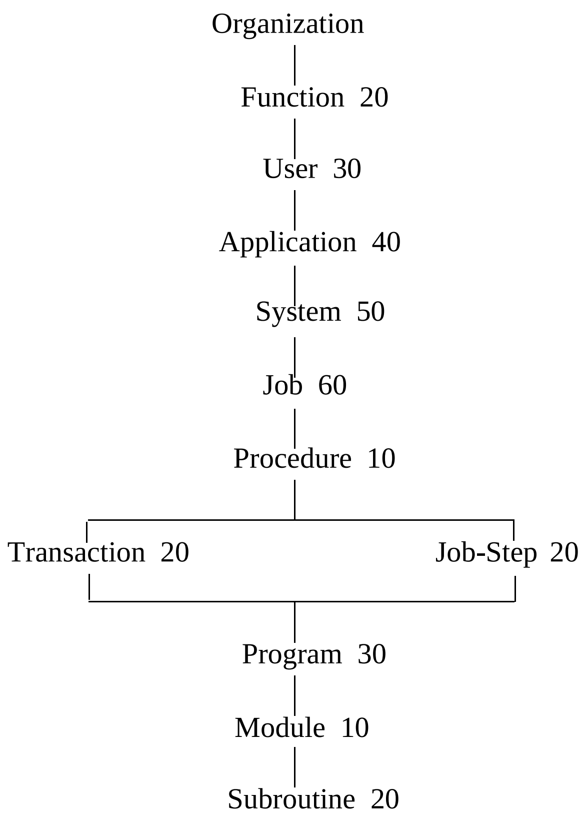
ASG also provides Danish, Dutch, Finnish, French, German, Italian, Norwegian, Spanish and Swedish versions of the EDPS structure. Each member type is given the equivalent name in the non-English language. For details see Appendix B, "Language Versions of the Extended Data Processing Structure," on page 111.

If you use a non-English language version of EDPS, then DataManager will still accept the English form of member type names (as well as the non-English form) on encoding or in member type selections in commands. However, references to member types in the output from LIST, GLOSSARY, and REPORT commands appear in the non-English language.

The EDPS Hierarchy

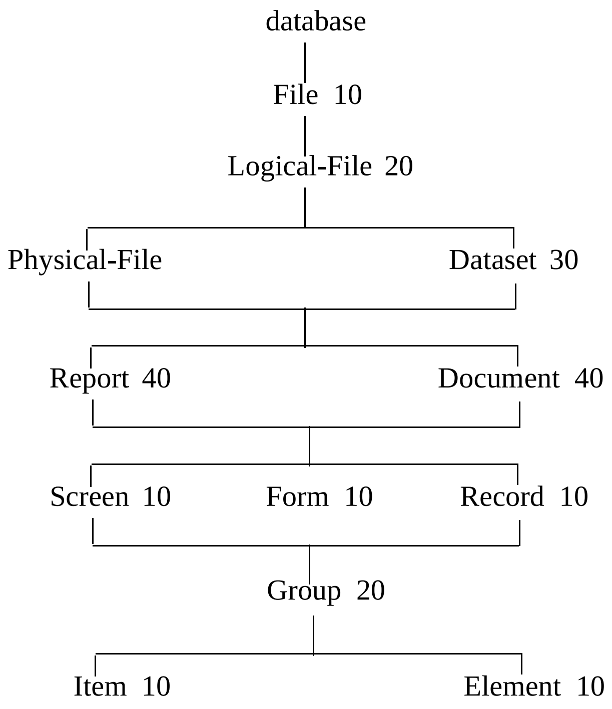
PROCESS Member Types

Figure 1 • EDPS Hierarchy (PROCESS member types)



DATA Member Types

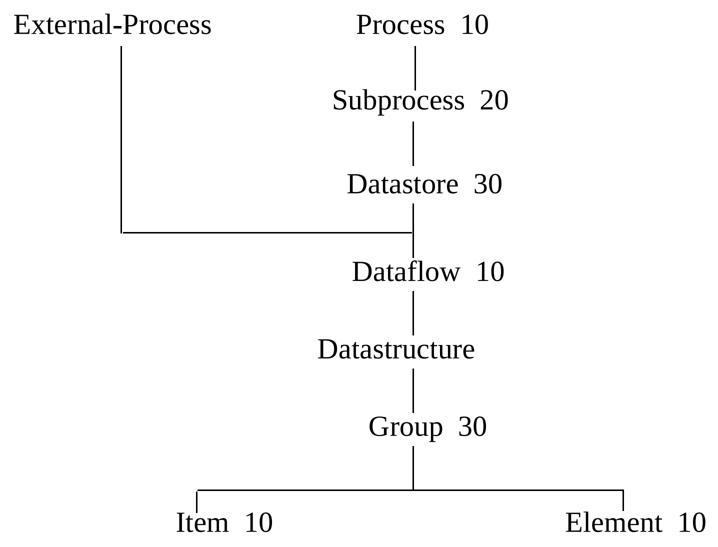
Figure 2 • EDPS Hierarchy (DATA member types)



The SAS Hierarchy

SAS (Figure 3) is a member type set that represents terms in top down or structured design techniques as promoted by Gane and Sarson of IST Inc., or by Yourdon Inc.

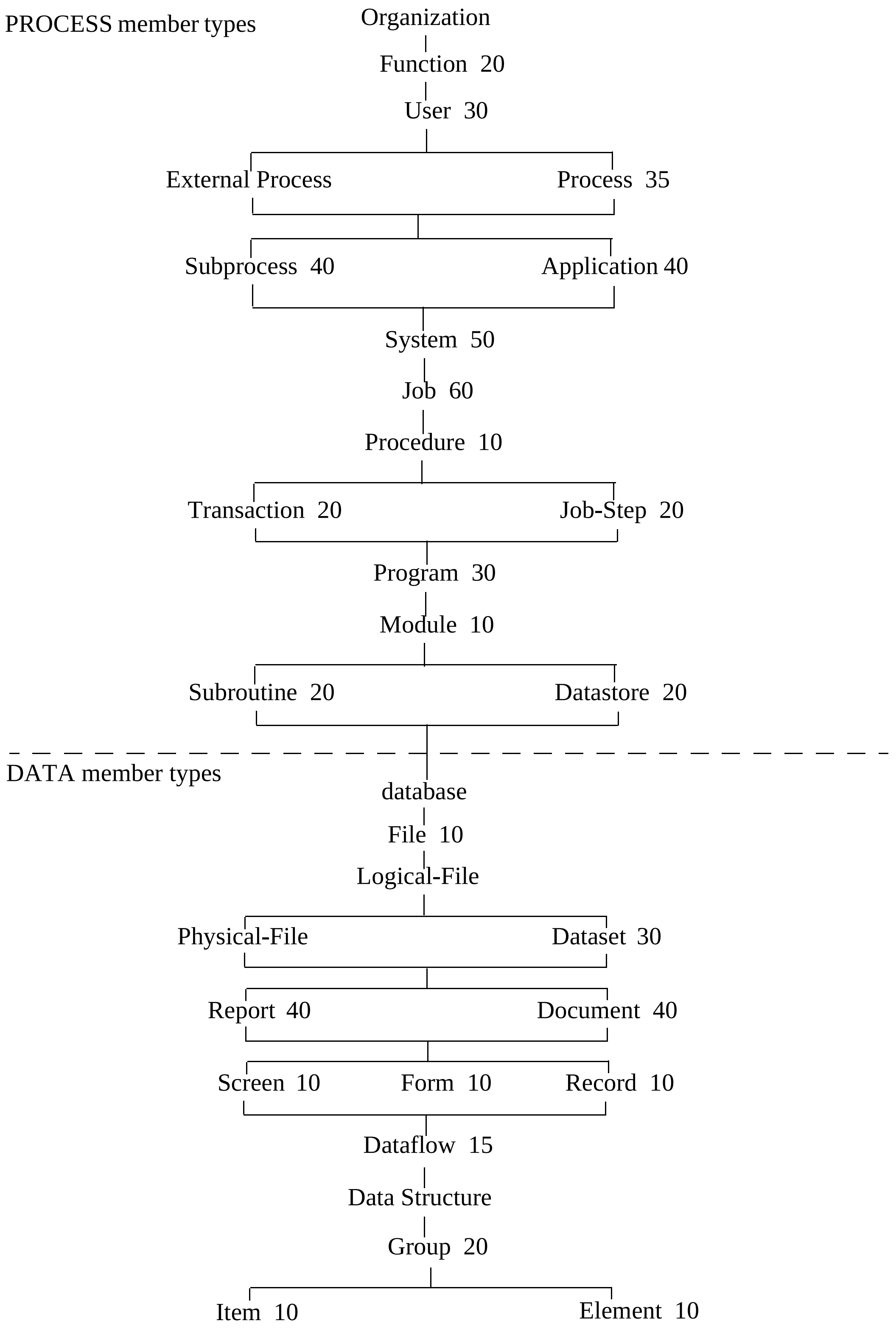
Figure 3 • SAS Hierarchy



The SDS Hierarchy

The SDS hierarchy shown in Figure 4 on page 8 is a combination of the EDPS and SAS member type hierarchies.

Figure 4 • SDS Hierarchy



2 Defining a Hierarchy Member

This chapter includes these sections:

The CONTAINS Clause

The CONTAINS clause may include the names of one or more MEMBER-TYPEs or MEMBER-TYPE-GROUPs separated by commas that are to form your hierarchy. If the clause contains the name of a MEMBER-TYPE-GROUP, all member types directly or indirectly contained in that group are included in the hierarchy. All member types in the hierarchy, including ASG-supplied member types, must be included directly or indirectly in the CONTAINS clause.

If, for example, you were defining the hierarchy of basic member types available with the DataManager Nucleus, it would look like this:

```
   HIERARCHY
   CONTAINS SYSTEM, PROGRAM, MODULE, FILE, GROUP, ITEM
 { : }
 { ; }
```

Each HIERARCHY member must contain one CONTAINS clause.

Refer to "UDS and IMS (DL/I) Member Types" on page 30 if your hierarchy includes IMS (DL/1) member types.

The MP-AID-NAME Clause of the HIERARCHY Member Type

If possible, the name of the UDS table should be the same as the name of the HIERARCHY member in the Manager Products Administration Dictionary. However, the UDS table is held on the MP-AID and the name of a UDS-TABLE member on the MP-AID can be no more than five characters long, whereas the name of a HIERARCHY member can be a maximum of 32 characters. If your HIERARCHY name is more than five characters long, therefore, you must specify an MP-AID name for the UDS table. This is the syntax of that clause:

```
MP-AID-NAME name
```

where *name* is a maximum of five characters, beginning with an alphabetic character and otherwise complying with the Manager Products naming standards.

If, for example, you had a HIERARCHY member named PROJECT-INTEGRATION you could specify an MP-AID name for it with this clause:

```
MP-AID-NAME PIH01
```

The COLLECTIVE Clause

The COLLECTIVE clause assigns a name to a set of MEMBER-TYPEs and/or MEMBER-TYPE-GROUPs. The name can be used in interrogation commands instead of referring individually to each of the MEMBER-TYPEs and MEMBER-TYPE-GROUPs in the set. For example, if you include this clause in the definition of the hierarchy:

```
COLLECTIVE FILES INCLUDES EGBFILE, ADFL,TMFL, TVFL
```

an interrogation that refers to FILES would generate details of members of every type in the collective set.

This is the syntax of the COLLECTIVE clause:

```
COLLECTIVE collective-name
     INCLUDES member-name [ , member-name ]...
```

where:

collective-name is the name that you give to the COLLECTIVE. May be a string of not more than 32 characters in the range A to Z, 0 to 9 and hyphen (-), beginning with a letter of the alphabet.

member-name is the name of a MEMBER-TYPE or MEMBER-TYPE-GROUP.

You can specify more than one COLLECTIVE clause in a hierarchy. Several COLLECTIVE clauses may refer to the same collective name, in which case all of the MEMBER-TYPEs and/or MEMBER-TYPE-GROUPs included in the second (or subsequent) COLLECTIVE clause will be added to the list for the first.

The COLLECTIVE clause is optional.

The COMMON-ATTRIBUTES Clause

The COMMON-ATTRIBUTES clause specifies those attributes which may be included in the definition of any memeber in the dictionary. For example, this clause:

```
COMMON-ATTRIBUTES EGBFRFM
```

specifies that the ATTRIBUTE-TYPE member EGBFRMF may be included in the definition of any member in the dictionary. The clause may contain the names of one or more ATTRIBUTE-TYPE or ATTRIBUTE-GROUP members separated by commas. If the clause refers to an ATTRIBUTE-GROUP, then all the attributes in the group are common to members of the dictionary.

Each name specified in the clause may be followed by an optional sub-clause which states whether or not the named attribute or the attributes in the named group need to be present when members in the hierarchy are encoded, as follows:

OPTIONAL YES	The attribute need not be present
OPTIONAL NO	The attribute must be present
OPTIONAL WARN	The attribute must be present but, if it is not, a warning message is issued

The default is OPTIONAL YES.

You can specify that ATTRIBUTE-TYPEs (but not ATTRIBUTE-GROUPs) are mutually exclusive by including an ELSE sub-clause, thus:

```
COMMON-ATTRIBUTES attribute-type
     [ELSE attribute-type]...
```

The ELSE sub-clause specifies that where one ATTRIBUTE-TYPE is present, the other must not be. For example, this clause:

```
COMMON-ATTRIBUTES EGBRRFM ELSE EGBKEY
```

indicates that members in the hierarchy may contain either the attribute defined by EGBRRFM or the attribute defined by EGBKEY, but not both. If two or more ELSE clauses are included, all ATTRIBUTE-TYPEs specified are mutually exclusive.

For any set of mutually exclusive ATTRIBUTE-TYPEs that you specify, an OPTIONAL sub-clause may follow the last ELSE sub-clause. OPTIONAL sub clauses may not be included for individual ATTRIBUTE-TYPEs named within the set.

This is the complete syntax of the COMMON-ATTRIBUTES clause:

```
COMMON-ATTRIBUTES attribute-clause
     [,attribute-clause]...
```

where *attribute-clause* is:

```
 { attribute-type [ ELSE attribute-type ]...  }
 { attribute-group                            }
   [OPTIONAL  { YES  } ]
              { NO   }
              { WARN }
```

The COMMON-ATTRIBUTES clause is optional.

The SYNONYM Clause

Each MEMBER-TYPE based on a base member type is assigned a numeric suffix which uniquely identifies that synonym of the base member type. The suffix can be assigned by default by CONTROLMANAGER when the UDS table based on the hierarchy is constructed to the MP-AID, or it can be explicitly specified by the SYNONYM clause in the HIERARCHY definition. The syntax of the clause is:

```
SYNONYM member-type-name IS BASED-ON base-member-type SUFFIX n
```

You may specify several SYNONYM clauses in a HIERARCHY definition, one for each MEMBER-TYPE in the hierarchy. *member-type-name* must be the name of a MEMBER-TYPE containing the BASED-ON and not the IS clause; see "The IS/BASED-ON Clause" on page 16.

When a UDS table is constructed, CONTROLMANAGER inserts in the HIERARCHY member definition a SYNONYM clause for each MEMBER-TYPE which has not already been explicitly assigned a suffix. So when the HIERARCHY member is printed, a SYNONYM clause will appear for every MEMBER-TYPE (other than MEMBER-TYPEs containing the IS clause) defined in the hierarchy.

If you update a HIERARCHY member, you are recommended to leave unchanged in the definition those SYNONYM clauses that relate to MEMBER-TYPEs that appear both in the old and in the new hierarchy. If you delete such SYNONYM clauses or assign a new suffix number to an existing MEMBER-TYPE, the MEMBER-TYPE in the new hierarchy may be treated as an entirely new MEMBER-TYPE, and a subsequent COMPARE UDS-TABLE command may fail.

These are example SYNONYM clauses from the definition of the HIERARCHY DU012 defined in the DEMO dictionary:

```
SYNONYM EGBAPPL IS BASED-ON SYSTEM SUFFIX 8
SYNONYM EGBJOB IS BASED-ON SYSTEM SUFFIX 9
SYNONYM EGBPROCD IS BASED-ON PROGRAM SUFFIX 1
SYNONYM EGBTRANS IS BASED-ON PROGRAM SUFFIX 2
SYNONYM EGBJOBST IS BASED-ON PROGRAM SUFFIX 3
SYNONYM EGBSUBR IS BASED-ON MODULE SUFFIX 1
```

The UDO Clause of the HIERARCHY Member Type

If the ControlManager User Defined Output facility (selectable unit CMR-UD15) is installed, each user defined attribute must be assigned a parameter number, line number, and format line number for the purposes of defining user-defined reports. The numbers are automatically generated by ControlManager when the UDS table is constructed onto the MP-AID or you may optionally specify the numbers yourself in the UDO clause of the HIERARCHY member type definition. This is the syntax of the clause:

```
UDO attribute-name IS PARAMETER-NUMBER n1 LINE n2
     [KEYWORD IS PARAMETER-NUMBER n3 [LINEn4]]
     FORHAT-LINE-NUMBER n5
```

where:

`attribute-name` must be the name of an ATTRIBUTE-TYPE.

`nl`, *`n2`*, *`n3`*, and *`n4`* are any integers in the range 1 to 32767.

`n5` is an integer which is a multiple of 3 in the range 2001 to 32767.

`nl`, *`n2`*, *`n3`*, *`n4`*, and *`n5`* must not be the same as the equivalent numbers for an existing user defined attribute.

The `KEYWORD` clause, not including `LINE` *`n4`*, must be included for any attribute except an UNIDENTIFIED KEYWORD attribute.

`LINE` *`n4`* may only be included if *`attribute-name`* identifies a TEXT attribute.

The numbers assigned to an attribute are output by the SHOW UDS command in any enquiry that outputs details of the attribute.

If you specify a UDO clause that is syntactically correct but does not conform to the rules given above, CONTROLMANAGER automatically generates a new clause, adding it to the end of the HIERARCHY definition. The numbers in the automatically generated clause are the numbers that must be used in a user defined report.

Syntax of HIERARCHY Member Type

```
HIERARCHY
  CONTAINS { member-type       }   [,{ member-type       } ]...
           { member-type-group }     { member-type-group }
  [MP-AID mpaid-name]
  [COLLECTIVE user-keyword
  INCLUDES { member-type       }   [,{ member-type       } ]...]...
           { member-type-group }     { member-type-group }
  [COMMON-ATTRIBUTES attribute-clause [,attribute-clause]...]
  [SYNONYM member-type IS BASED-ON base-member-type SUFFIX n]...
  [UDO attribute-name IS PARAMETER-NUMBER n1 LINE n2
       [KEYWORD IS PARAMETER-NUMBER n3 [LINE n4]]
        FORMAT-LINE-NUMBER n5]...
  [common clauses]
{ . }
{ ; }
```

where:

`attribute-clause` is:

```
{ attribute-type [ ELSE attribute-type ]... } [OPTIONAL  { YES  } ]
{ attribute-group                           }            { NO   }
                                                         { WARN }
```

`attribute-type` must be the name of an ATTRIBUTE-TYPE

`n1`, `n2`, `n3`, and `n4` are any integers in the range 1 to 32767

`n5` is an integer that is a multiple of 3 in the range 2001 to 32767

`n1`, `n2`, `n3`, `n4`, and `n5` must not be the same as the equivalent numbers for any existing user-defined attribute

`common-clauses` are as defined in Chapter 9, "Common Clauses," on page 63.

3 Defining a Member-type Member

This chapter includes these sections:

The IS/BASED-ON Clause

Each MEMBER-TYPE in your hierarchy must be modeled on one of the ASG-supplied member types. Any ASG-supplied member type available at your installation [except IMS (DL/I) member types] may be used as a base member for user-defined member types. The relationship clauses in the base member type may be included in members of the user-defined member type. Special rules apply to IMS (DL/I) member types. [For details, see "UDS and IMS (DL/I) Member Types" on page 30.]

Several MEMBER-TYPEs may be modeled on a single ASG-supplied member type. You are strongly recommended to indicate that one of the MEMBER-TYPEs modeled on a given ASG- supplied member type is the equivalent of that member type by including this clause in its definition:

```
IS member-type
```

for example:

```
IS MODULE
```

Only one of the MEMBER-TYPEs modeled on a given ASG-supplied member type may contain the IS clause. For example, only one MEMBER-TYPE may contain this clause:

```
IS MODULE
```

All the other MEMBER-TYPEs modeled on that member type must contain this clause:

```
BASED-ON member-type
```

for example:

```
BASED-ON MODULE
```

Members that are defined to be BASED-ON another MEMBER-TYPE may not contain the REPORT-DOWN-TO-KEYWORDS or GENERIC-ATTRIBUTES clauses. Otherwise the definitions of IS and BASED-ON MEMBER-TYPEs may contain the same clauses.

All members modeled on an ASG-supplied member type may be selected by the GENERIC keyword.

Relationships Between Members

A member of a user-defined member type may contain any of the relationship clauses of the member type on which it is modeled (known as the base member type). For example a member modeled on an ITEM member type may contain the USER-EXIT, NAME, IF, or SEE clauses.

By default a relationship clause in a member of a user-defined member type may refer to members:

- Of any of the types to which the clause can refer in the base member type, or
- Based on any of such types.

For example, the USER-EXIT clause in an ITEM member, or a member based on an ITEM member, may refer to a MODULE or REUSABLE-CODE member or to any member based on a MODULE or REUSABLE-CODE member.

You cannot extend the relationships that are permitted by default but you can restrict them by including these clauses in the definition of a member type:

- The RECURSIVE clause specifies that a member may or may not refer to members of the same type.
- The LEVEL clause specifies a hierarchy within member types based on the same base member type. Members with a higher level number are usually unable to refer to members with the same or a lower level number.
- The RELATIONSHIPS VIA clause allows or disallows references to other member types via a specified clause. The member type must fall within the range of member types to which reference may be made by default. The RELATIONSHIPS VIA clause also lets you specify the type of dummy member created if a clause refers to a member that does not exist.

These clauses are discussed in "The RECURSIVE Clause" on page 18, "The LEVEL Clause" on page 18 and "The RELATIONSHIPS VIA Clause" on page 19, respectively. Relationships permitted from base member types are summarized in Appendix A, "Relationships Permitted from Base Member Types," on page 67.

The RECURSIVE Clause

By default members of a given type may refer to each other. To specify that members of a given type may not refer to each other, include in the MEMBER-TYPE definition the clause:

```
RECURSIVE NO
```

RECURSIVE YES may also be included for documentary purposes.

You may override the effect of the RECURSIVE clause for any particular clause of a member definition by means of the RELATIONSHIPS VIA clause.

The LEVEL Clause

By default a relationship clause in a user-defined member may refer to members:

- Of any of the types to which the clause can refer in the base member type, or
- Based on any of such types.

For example, the NAME clause in a member based on an ITEM may refer to an ITEM or any member based on an ITEM.

If the LEVEL clause is included in the definition of a MEMBER-TYPE, members of that type may refer only to:

- Members based on the same base member type with a higher level number, or
- Other members in the hierarchy to which reference is permitted as stated above.

An exception to this rule is given below.

The syntax of the LEVEL clause is:

```
LEVEL n
```

where *n* is an unsigned integer in the range 0 to 255.

For example, the following diagram of the SAS hierarchy shows the user-defined member types PROCESS, SUBPROCESS and DATASTORE, all based on SYSTEM (which itself is not a part of the SAS hierarchy). Within this grouping the level numbers dictate that PROCESS members may refer to SUBPROCESS members which may refer to DATASTORE members, but that DATASTORE members may not refer to SUBPROCESS or PROCESS-members, and SUBPROCESS members may not refer to PROCESS members.

```
PROCESS 10        \
                   |
SUBPROCESS 20      >        all based on SYSTEM
                   |
DATASTORE 30      /
```

Two or more user-defined member types which share the same base member type may have the same level number but in such a case they are not able to refer to each other.

If a MEMBER-TYPE is given a level number of 0, members of that type may refer to other members whose MEMBER-TYPE has a level number of 0.

You may override the effect of the LEVEL clause for any particular clause of a member definition by means of the RELATIONSHIPS VIA clause.

The RELATIONSHIPS VIA Clause

The RELATIONSHIPS VIA clause is optional. The clause lets you modify the relationships that would otherwise be permitted from specified clauses of a member definition.

Disallowing Relationships

To disallow references from a specified clause, include in the definition of the member type:

RELATIONSHIPS VIA *clause* DISALLOWED

For example:

```
RELATIONSHIPS VIA CALLS DISALLOWED
```

disallows the use of the CALLS clause in members of the type being defined.

Allowing Relationships

To define the relationships permitted from a specified clause, include:

```
RELATIONSHIPS VIA keyword
ALLOW ⎧member-type       ⎫   [,⎧ member-type       ⎫ ]...
      ⎩member-type-group ⎭     ⎩ member-type-group ⎭
```

where:

`keyword` is the name of the clause

`member-type` is the name of the MEMBER-TYPE member of your Manager Products Administration Dictionary that defines the member type to which reference is allowed.

If `member-type-group` is specified, then reference is allowed to all MEMBER-TYPEs in the group. For example, this clause:

```
RELATIONSHIPS VIA CONTAINS ALLOW SASITEM, UIDMS
```

allows the CONTAINS clause to refer to the MEMBER-TYPE SASITEM and to all the MEMBER-TYPEs in the group UIDMS. The relationship must be within the relationships that would be permitted from the MEMBER-TYPE being defined (disregarding the effect of any RECURSIVE or LEVEL clause). References from `keyword` may only be to the MEMBER-TYPEs specified in the ALLOW sub-clause.

Defining the Type of Dummy Members

If, when it is encoded, a member of the type being defined refers to a member that does not already exist in the data entries data set, then a dummy member is created. The type of dummy member that is created depends on the type of member and the clause from which the reference is made.

To change the default when the RELATIONSHIPS VIA clause is not otherwise specified in the MEMBER-TYPE definition, include:

```
RELATIONSHIPS VIA keyword DUMMY member-type
```

where `member-type` can be any MEMBER-TYPE to which reference could be made via keyword from members of the type being defined. For example, this clause:

```
RELATIONSHIPS VIA CONTAINS DUMMY EGBGROUP
```

causes a dummy EGBGROUP member to be created if reference is made in the CONTAINS clause to a member that does not exist in the data entries data set.

Where the relationship is allowed by the RELATIONSHIPS VIA clause, then a dummy of the first MEMBER-TYPE named in the clause is created. For example, if this clause:

```
RELATIONSHIPS VIA IF ALLOW SASITEM, ELEM
```

is included, then a dummy SASITEM member would be created. If the first name in the RELATIONSHIPS VIA clause is the name of a MEMBER-TYPE-GROUP, then a dummy is created of the first type in the group.

If you want a dummy of a different MEMBER-TYPE to be created, then you can explicitly specify the type of dummy by including the DUMMY clause. The name specified in the DUMMY clause can be the name of a MEMBER-TYPE within a MEMBER-TYPE-GROUP named in the clause, for example:

```
RELATIONSHIPS VIA KEYS ALLOW EGB-UBASEU DUMMY EGBPROG
```

where `EGBPROG` is a MEMBER-TYPE contained within the MEMBER-TYPE-GROUP EGB-UBASEU.

Defining Your Member Type's Attributes

A member may contain these attributes:

- Common clauses
- Attributes included in the COMMON-ATTRIBUTES clause of the HIERARCHY to which the member belongs
- Attributes included in the GENERIC-ATTRIBUTES clause of the MEMBER-TYPE on which the member is based
- Attributes included in the ATTRIBUTES clause of the MEMBER-TYPE definition of the member.

Note that in addition to attributes, a member may also contain relationship clauses.

GENERIC-ATTRIBUTES *member-name* [, *member-name*]...

Common clauses are as shown in "Syntax of MEMBER-TYPE Member Type" on page 28. For COMMON-ATTRIBUTES, see "The COMMON-ATTRIBUTES Clause" on page 11. For GENERIC-ATTRIBUTES, see "The GENERIC-ATTRIBUTES Clause" on page 22. The ATTRIBUTES clause is discussed in "The ATTRIBUTES Clause" on page 23.

The GENERIC-ATTRIBUTES Clause

The GENERIC-ATTRIBUTES clause may only be included in the definition of a MEMBER-TYPE that contains the IS rather than the BASED-ON clause. Attributes named in the GENERIC-ATTRIBUTES clause, or attributes in the ATTRIBUTE-GROUPs named in the clause, may be included in the definition of any member based on the same base member type. This is the syntax of the clause:

```
GENERIC-ATTRIBUTES member-name [, member-name]...
```

where `member-name` is the name of an ATTRIBUTE-TYPE or an ATTRIBUTE-GROUP.

Each name specified in the clause may be followed by an OPTIONAL sub-clause which states whether or not the named attribute, or the attributes in the named group, need to be present when members in the hierarchy are encoded, as follows:

OPTIONAL YES	The attribute need not be present.
OPTIONAL NO	The attribute must be present.
OPTIONAL WARN	The attribute need not be present but, if it is not, a warning message is issued.

The default is OPTIONAL YES.

You can specify that ATTRIBUTE-TYPES (but not ATTRIBUTE-GROUPs) are mutually exclusive by including an ELSE clause, thus:

```
GENERIC-ATTRIBUTES attribute-type
     [ELSE attribute-type]...
```

The ELSE sub-clause specifies that where one ATTRIBUTE-TYPE is present, the other must not be. For example:

```
GENERIC-ATTRIBUTES EGBFRFM ELSE EGBKEY
```

indicates that members in the hierarchy may either contain the attribute EGBFRFM or the attribute EGBKEY. If two or more ELSE clauses are included, each ATTRIBUTE-TYPE specified is mutually exclusive.

The ATTRIBUTES Clause

The ATTRIBUTES clause specifies attributes that may be included in members of the type being defined. This is the syntax of the clause:

```
ATTRIBUTES member-name [, member-name]...
```

where `member-name` is the name of an ATTRIBUTE-TYPE or ATTRIBUTE- GROUP. If the clause refers to an ATTRIBUTE-GROUP then any of the attributes in the group may be included in the definition of members of the type being defined.

Each name specified in the clause may be followed by an OPTIONAL sub-clause which states whether or not the named attribute, or the attributes in the named group, need to be present when members in the hierarchy are encoded, as follows:

OPTIONAL YES	The attribute need not be present.
OPTIONAL NO	The attribute must be present.
OPTIONAL WARN	The attribute need not be present but, if it is not, a warning message is issued.

The default is OPTIONAL YES.

You can specify that ATTRIBUTE-TYPEs (but not ATTRIBUTE-GROUPs) are mutually exclusive by including an ELSE clause, thus:

```
ATTRIBUTES attribute-type [ELSE attribute-type]...
```

The ELSE sub-clause specifies that where one ATTRIBUTE-TYPE is present, the other must not be. For example:

```
ATTRIBUTES EGBFRFM ELSE EGBKEY
```

indicates that members in the hierarchy may either contain the attribute EGBFRFM or the attribute EGBKEY. If two or more ELSE clauses are included, each ATTRIBUTE-TYPE specified is mutually exclusive.

Defining the Names of your Member Type

You may specify different keywords to be used to identify the member type you are defining, in:

- The definition statements of members of that type. These member type identifier keywords are defined as ENCODE- KEYWORDS.
- Interrogation commands. These are defined as INTERROGATE-KEYWORDS.
- The DOWN-TO clause of the REPORT command. These are defined as REPORT-DOWN-TO-KEYWORDS.

Defining Your Member Type's ENCODE-KEYWORDS

The name given to a MEMBER-TYPE in the dictionary does not need to be the same as the keyword used for members of that type when they are encoded. The ENCODE-KEYWORDS clause defines the keywords to be used for members of that type when they are encoded. For example, the definition of the MEMBER-TYPE EGBELEM in the DEMO dictionary contains these clauses:

```
MEMBER-TYPE
BASED-ON ITEM
LEVEL 10
ENCODE-KEYWORDS ELEMENT
...
```

So when a member of this type in encoded, its definition must begin with the keyword ELEMENT, for example:

```
ELEMENT
DESCRIPTION 'Production code'
...
```

This is the syntax of the ENCODE-KEYWORDS clause:

ENCODE-KEYHORDS *user-keyword* [, *user-keyword*]...

user-keyword must obey the rules for undelimited member names and must begin with a letter of the alphabet. The keywords must not be the same as the ENCODE-KEYWORDS of any other MEMBER-TYPE in the hierarchy. Where several keywords are specified, members of that type may be encoded under any of the keywords.

Certain database MEMBER-TYPEs use a two-level keyword system which is affected if you change the ENCODE-KEYWORDS of their base member type.

Thus MARK IV, TOTAL, or ADABAS file member types use a two level keyword system when encoding. This means that if you change the ENCODE-KEYWORDS of the FILE member type you must use the new ENCODE-KEYWORDS to encode FILE members of these types. Thus, where you have the member type FILE TOTAL and you have changed the ENCODE-KEYWORDS of FILE to CASE, then you would use the keyword CASE TOTAL to encode members of that type.

The ENCODE-KEYWORDS clause is optional but, if it is omitted, it is not possible to encode members of the type being defined. Thus, if you want your HIERARCHY to continue to include existing members of an obsolescent type, but do not want to accept any new members of of that type, you can:

- Remove the ENCODE-KEYWORDS clause from your MEMBER-TYPE member
- Change the mpaid-name in the MP-AID clause of your HIERARCHY member
- Construct a new UDS table and apply it to your dictionary.

If the ENCODE-KEYWORDS clause is omitted, then you must include an INTERROGATE-KEYWORDS clause (which would otherwise be optional) in your MEMBER-TYPE definition.

If the ENCODE-KEYWORDS clause is omitted, affected members would not re- encode if you RESTORE your dictionary or perform any other operations that involve a bulk re-encoding.

Defining Your Member Type's INTERROGATE-KEYWORDS

When you interrogate the dictionary, you can select the members about which information is to be provided by member type. For example:

```
LIST PROGRAMS;
```

lists details of all programs on the dictionary. PROGRAMS is known as an interrogate keyword .

The INTERROGATE-KEYWORDS clause specifies the interrogate keywords for a member type. This is the syntax of the clause:

```
INTERR0GATE-KEYWORDS user-keyword [, user-keyword]...
```

`user-keyword` must obey the rules for undelimited member names and must begin with a letter of the alphabet. *`user-keyword`* must not be the same as the interrogate keyword of any other MEMBER-TYPE in the hierarchy.

The INTERROGATE-KEYWORDS clause is optional. If it is omitted, ControlManager inserts an INTERROGATE-KEYWORDS clause in the definition of the MEMBER-TYPE when the UDS table is constructed to the MP-AID. The default keywords are the keywords in the ENCODE-KEYWORDS clause followed by S, or ES when the keyword ends in S, SH, CH or X. For example BATCHES is the plural form of BATCH.

For example, assume that the MEMBER-TYPE definition contains this clause:

```
ENCODE-KEYWORDS CONVENTIONAL-FILE
```

and no INTERROGATE-KEYWORDS clause; then an interrogation that refers to CONVENTIONAL-FILES would include details of members of the member-type being defined, for example:

```
WHICH CONVENTIONAL-FILES DIRECTLY USE EGBGROUP;
```

The REPORT-DOWN-TO-KEYWORDS Clause

This is one version of the REPORT command:

```
REPORT member-name DOWN-TO member-type;
```

The command issues a report on the named member and all members directly and indirectly referenced by that member down to the level of the member type specified.

The keyword used to specify the member type in the REPORT...DOWN-TO command is by default the interrogate keyword of the MEMBER-TYPE.

However, in the case of a MEMBER-TYPE whose definition contains the IS rather than the BASED-ON clause, you can specify a different keyword for the MEMBER-TYPE for use in the REPORT...DOWN-TO command. The keyword is specified by means of the REPORT-DOWN-TO-KEYWORDS clause. This is the syntax of the REPORT-DOWN-TO-KEYWORDS clause is:

```
REPORT-DOWN-TO-KEYWORDS
     user-keyword [, user-keyword]...
```

where *user-keyword* must obey the rules for undelimited member names and must begin with a letter of the alphabet. The keywords must not be the same as the encode keywords or interrogate keywords of any other MEMBER-TYPE in the hierarchy. Where several REPORT-DOWN-TO-KEYWORDS are specified, any of the keywords may be used to refer to the MEMBER-TYPE in the REPORT...DOWN-TO command. A MEMBER-TYPE which is defined as being BASED-ON another member type may be referred to in a REPORT...DOWN-TO command, but in such a case it must be referenced by one of its interrogate keywords.

The REPORT...DOWN-TO command traces all reference paths from the named member down to and including members of the named MEMBER-TYPE. The path includes members based on the same MEMBER-TYPE as the MEMBER TYPE referenced in the command, provided that their level numbers are lower than the level number of that MEMBER-TYPE.

Defining Your Member Type's LITERAL Values

A MEMBER-TYPE's literal values are the names by which output from the interrogation commands refer to the MEMBER-TYPE. There are four LITERAL clauses. The keywords of the clauses are given below followed by the names of the commands whose output uses the literal in each case.

Keyword	**Commands**
STANDARD-LITERAL	The REPORT, WHAT, WHICH, and GLOSSARYcommands.
SHORT-LITERAL	The LIST command.
PLURAL-LITERAL	Headings and totals of output from the GLOSSARY and LIST commands.
LONG-LITERAL	Included for IMS (DL/I) MEMBER-TYPEs for message and SHOW command output.

Each keyword may be followed by a delimited string. In the case of the STANDARD-LITERAL, PLURAL-LlTERAL and LONG-LITERAL clauses, the string may be not more than 32 characters long. In the SHORT-LITERAL clause, the string must be not more than 12 characters long.

The STANDARD-LITERAL clause must be included in every MEMBER-TYPE definition. The default values for the other clauses are given below.

Clause	**Default**
SHORT-LITERAL	The STANDARD-LITERAL if the STANDARD-LITERAL is not longer than 12 characters. If the STANDARD-LITERAL is longer than 12 characters and no SHORT-LITERAL is defined, an error occurs.
PLURAL-LITERAL	The STANDARD-LITERAL followed by S, unless the literal ends in S, SH, CH or X, in which case it is the STANDARD-LITERAL followed by ES; for example BATCH would be extended to BATCHES.
LONG-LITERAL	The STANDARD-LITERAL.

Note that if the PLURAL-LITERAL is defaulted, the literal in the STANDARD-LITERAL clause, when extended by S or ES, should not exceed 32 characters.

Example

If a MEMBER-TYPE definition includes these clauses:

```
STANDARD-LITERAL 'PHYSICAL FILE'
SHORT-LITERAL 'PHYS FILE'
```

then output from the interrogation commands would refer to the member type as PHYSICAL FILE, except for the LIST command which would refer to it as PHYS FILE. The headings and totals of GLOSSARY and LIST output would refer to the members of this type as PHYSICAL FILES.

Syntax of MEMBER-TYPE Member Type

```
MEMBER-TYPE
  ⎧ IS       ⎫ base-member-type
  ⎩ BASED-ON ⎭

   [LEVEL n]
   [RECURSIVE ⎧ YES ⎫ ]
              ⎩ NO  ⎭
   [ENCODE-KEYWORDS user-keyword [, user-keyword]...]
   [INTERROGATE-KEYWORDS user-keyword [, user-keyword]...]
   [REPORT-DOWN-TO-KEYWORDS user-keyword [, user-keyword]...]
     STANDARD-LITERAL 'string'
   [SHORT-LITERAL 'string']
   [PLURAL-LITERAL 'string']
   [LONG-LITERAL 'string']
   [RELATIONSHIPS VIA relationship-keyword
     ⎧ DISALLOWED                                          ⎫ ]...
     ⎩ ALLOW member-type-clause][DUMMY member-type]        ⎭
   [ATTRIBUTES attribute-clause [, attribute-clause]...]
   [GENERIC-ATTRIBUTES attribute-clause [, attribute-clause]...]
   [common-clauses]
⎧ . ⎫
⎩ ; ⎭
```

where:

member-type-clause is:

```
⎧ member-type       ⎫ [, ⎧ member-type       ⎫]...
⎩ member-type-group ⎭    ⎩ member-type-group ⎭
```

attribute-clause is:

```
⎧ attribute-type [ ELSE attribute-type ]... ⎫ [OPTIONAL   ⎧ YES  ⎫ ]
⎨                                           ⎬             ⎨ NO   ⎬
⎩ attribute-group                           ⎭             ⎩ WARN ⎭
```

relationship-keyword for any given member type may be any of the keywords marked with an X in this table:

Keyword	**Member Type**					
	File	**Group**	**Item**	**System**	**Program**	**Module**
CONTAINS	X	X				
IF	X	X				
USER-LABELS	X					
SORT-KEY	X					
KEYS		X				
USER-EXIT		X	X			
NAME			X			
IF (of CONTAINS)			X			
SEE	X	X	X	X	X	X
INPUTS				X	X	X
OUTPUTS				X	X	X
UPDATES				X	X	X
CALLS				X	X	X
PASSES				X	X	X
PARAMETERS				X	X	X

common-clauses are as defined in Chapter 9, "Common Clauses," on page 63.

The GENERIC Keyword

Commands that permit the selection of members by member type may include the GENERIC keyword, for example:

```
LIST GENERIC GROUPS
```

The GENERIC keyword may be followed by the interrogate keyword of any member type in your hierarchy whose definition contains the IS clause. The keyword selects all members of that type and all members whose member type is modelled on the member type referred to in the IS clause.

See "The IS/BASED-ON Clause" on page 16 for a description of the IS/BASED-ON clause.

UDS and IMS (DL/I) Member Types

You cannot base user-defined member types on IMS member types. However, you can include IMS member types in your hierarchy by defining them in MEMBER-TYPE definitions containing the IS clause and any of the other clauses permitted in such definitions.

IMS member type defining databases, segments, and PCBs can be identified at two levels. The first level identifies the member type as an IMS-DATABASE, a SEGMENT or a PCB. The second level, which in dictionary members of any of these types is specified by the second keyword in the definition, identifies the particular type of IMS database, segment or PCB; as, for example, in IMS-DATABASE GSAM.

The first level identifier for these IMS member types can be renamed by specifying an ENCODE- KEYWORD, thus:

```
MEMBER-TYPE IS IMS-DATABASE
ENCODE-KEYWORD DATABASE
```

This allows you to encode a member whose source record starts with DATABASE GSAM, for example. The second level identifier (GSAM in this example) of members defined in this way cannot be renamed.

You must specify a MEMBER-TYPE member for each of the particular types of IMS-DATABASE, SEGMENT and PCB that you wish to include in your hierarchy. These correspond to the second level of identification mentioned above. You can optionally specify a single ENCODE-KEYWORD for the member type at this level; for example,

```
MEMBER-TYPE IS IMS-GSAM-DATABASE
ENCODE-KEYWORD GSAM-DATABASE
```

If an ENCODE-KEYWORD is not specified for a member type at this level, then members of that member type must be defined using both levels of keyword, as in IMS-DATABASE GSAM.

For a list of all the second level IMS member types, together with the keywords available for use in their RELATIONSHIPS VIA clause, see Appendix C, "Relationships Permitted from IMS Member Types," on page 133.

If your hierarchy contained MEMBER-TYPE definitions at both levels as in the examples above, then you could encode a member defining a GSAM IMS database whose source record started with either DATABASE GSAM or GSAM-DATABASE.

Whereas a second level MEMBER-TYPE definition must be included in your hierarchy for each relevant IMS member type, it is not essential to include first level definitions. ControlManager will generate default first level IMS-DATABASE, SEGMENT, and PCB entries in your UDS table from your second level definitions if you do not specify them yourself. Thus, HIERARCHY members do not have to CONTAIN first level members (unless you wish to rename them or to specify user-defined attributes for them) but may do so. For completeness of documentation, ASG recommends that such members be included.

Internal IMS member types (for example, CONCATENATED-KEY-FIELD) can also be specified and included in your hierarchy. This enables you to change their literals. The ENCODE-KEYWORDS clause should be omitted, because members of these types do not have source records. If you do not specify the internal IMS member types, default entries for them will be included in your UDS table.

Interrogate keywords

Interrogate keywords defined in a first level IMS member type definition will, when used in an appropriate command, select all members of the specified type, regardless of their second level keyword.

Interrogate keywords defined in a first level IMS member type definition will, when used in an appropriate command, confine the selection to members of the specified second level type.

Literals

Literals for individual members appearing in the output from commands such as LIST or GLOSSARY will be those specified in the second level definitions. Literals appearing for the totals in the output from these commands will be those specified in the first level definitions; these totals are accumulated by the first level of member type only.

Attributes

An ATTRIBUTES clause should be included only in first level definitions. If included in a second level definition, a warning message will be output when the UDS table is constructed, and the clause will be otherwise ignored.

It is not meaningful to include a GENERIC-ATTRIBUTES clause in an IMS member type definition, because other member types cannot be based on an IMS member type.

4 Defining a Member-type-group Member

A MEMBER-TYPE-GROUP groups MEMBER-TYPEs and MEMBER-TYPE-GROUPS together so that they can be referred to by a single name.

This is the syntax of MEMBER-TYPE-GROUP:

```
MEMBER-TYPE-GROUP
   CONTAINS member-name [, member-name]...
   common-clauses
{ : }
{ ; }
```

where:

member-name is the name of a MEMBER-TYPE or MEMBER-TYPE-GROUP

common-clauses are as defined in Chapter 9, "Common Clauses," on page 63.

Example

```
MEMBER-TYPE-GROUP
CONTAINS UADA, UTOT, US2X, UMIV, UIDMS
;
```

5 Defining an Attribute Type

This chapter includes these sections:

Introduction

You may define your own attributes for inclusion either in ASG-supplied or user-defined member types. An attribute consists of a keyword that identifies the attribute (except in the case of an unidentified KEYWORD attribute) and a value. For example, in this attribute:

```
DESCRIPTION "Employee identification"
```

`DESCRIPTION` is the keyword and `Employee identification` the value.

The value that may be given to an attribute in a member definition depends on the type of the attribute. These attribute types are available:

- CHARACTER-STRING
- DATE
- DECIMAL-NUMBER
- FREE-FORM-TEXT
- INTEGER
- KEYWORD
- NAME
- TEXT
- TIME

They are described in "The DATE Attribute Type" on page 36 to "The TEXT Attribute Type" on page 53, respectively.

"Indexing User Defined Attributes" on page 54 describes the optional indexing of user-defined attributes.

For the common-clauses available in ATTRIBUTE-TYPE members (as in all member types) refer to the *ASG-Manager Products Dictionary/Repository User's Guide*.

The DATE Attribute Type

Defining a DATE Attribute

The value of a DATE attribute in a member containing that attribute must be a date, for example `31 JAN 95`. The date must be in the format defined in the DCUST installation macro.

To define a DATE attribute, enter this member definition:

```
ATTRIBUTE-TYPE
DATE
IDENTIFIED-BY user-keyword
[optional-clauses]
;
```

where:

`user-keyword` is the name by which the attribute is referred to in definitions of members containing that attribute. The keyword must obey the rules for undelimited member names and must not begin with a numeral. It must not be the same as the keyword identifying any other attribute in the UDS table.

`optional-clauses` may be any of the common clauses and/or any of the clauses described below.

To specify that the attribute is to be indexed by presence or value, you may optionally include in the definition:

```
INDEXED-BY PRESENCE
```

or

```
INDEXED-BY VALUE
```

If indexed by value, an encoded member having this attribute is indexed by the normalized value (see below) irrespective of the format in which the value is entered.

To specify the permitted values of the attribute, you may optionally include in the definition:

`VALUES` *`date`*

where *`date`* must be in the format in use at your installation. A member containing an attribute whose value is not one of the permitted values will then not encode successfully. You can specify any number of dates. If you do not know what that format is, you can alternatively include in the definition:

`NORMALIZED-VALUES` *`yyyyddd`*

where *`yyyy`* is the year and *`ddd`* is the number of the day within the year (in the range 001 through 366).

Instead of specifying pre-defined values for the attribute, you may specify the maximum and/or minimum value of the attribute. To do this, include in the definition:

`MINIMUM-VALUE` *`date`* `MAXIMUM-VALUE` *`date`*

where *`date`* must be in the format in use at your installation. You can alternatively specify:

`NORMALIZED-MINIMUM-VALUE` *`yyyyddd`* `NORMALIZED-MAXIMUM-VALUE` *`yyyyddd`*

where *`yyyyddd`* is as described above.

You can specify that more than one value is required in an attribute for successful encoding by including this attribute:

```
MULTIPLE-VALUES
```

in the definition.

You may also specify either, or both, the minimum and maximum number of values permitted in an attribute, by including in the definition:

```
MULTIPLE-VALUES MINIMUM-NUMBER integer
                MAXIMUM-NUMBER integer
```

where `integer` is an integer in the range 1 through 32767.

A member containing an attribute with too few and/or too many values will then not encode successfully.

Example

```
ATTRIBUTE-TYPE
DATE
IDENTIFIED-BY PERIOD
INDEXED-BY PRESENCE
NORMALIZED-MINIMUM-VALUE 1984001
NORMALIZED-MAXIMUM-VALUE 1964366
MULTIPLE-VALUES MAXIMUM-NUMBER 3
;
```

Permitted Format for a DATE Value

If the DCUST installation macro has been tailored, the date must be in the format defined in the macro. Please consult your Systems Administrator or repository Controller if you are not sure what the format is.

Each element of the date must be separated by any printable character or space. If the separator is a space, comma, semi-colon or right round bracket, the date must be enclosed in delimiters. Otherwise delimiters are optional. If the macro was not tailored, this is the format of the date:

day month year

where:

day may be one or two numeric digits.

month may be either one or two numeric digits, or one of these character strings (unless tailored) which may be unambiguously truncated:

JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC

year may be two digits or four digits in the range 1000 through 2999.

The DECIMAL-NUMBER Attribute Type

The value of a DECIMAL-NUMBER attribute in a member containing that attribute may be a maximum of 31 digits, optionally preceded by a sign and/or containing a decimal point. The decimal point must not be the final character.

To define a DECIMAL-NUMBER attribute, enter this member definition:

```
ATTRIBUTE-TYPE
DECIMAL-NUMBER
IDENTIFIED-BY user-keyword
[optional-clauses]
;
```

where:

> `user-keyword` is the name by which the attribute is referred to in definitions of members containing that attribute. The keyword must obey the rules for undelimited member names and must not begin with a numeral. It must not be the same as the keyword identifying any other attribute in the UDS table.
>
> `optional-clauses` may be any of the common clauses and/or any of the clauses described below.

To specify that the attribute is to be indexed, you may optionally include in the definition:

```
INDEXED-BY PRESENCE
```

or

```
INDEXED-BY VALUE
```

The attribute cannot be indexed by value if its maximum length is greater than 18 digits. If the attribute is indexed by value, it is indexed with leading and trailing zeros to 18 digits before the decimal point and 18 places after the decimal point, and a sign.

For example, 1.25 is indexed as + 000000000000000001.250000000000000000.

This is to ensure that interrogations such as:

```
WHICH MEMBERS HAVE attribute EQUALS 1.25;
```

give the correct numeric value results (so that, for example, 1.25, 001.25, +1.25 and 1.2500 are not treated as different numbers).

To limit the number of digits that may appeal in the attribute value, you may optionally include in the definition of the attribute:

```
MINIMUM-LENGTH n
```

and/or

```
MAXIMUM-LENGTH n
```

where *n* is an integer not greater than 31. If no maximum length is specified, then a length of 18 is used as the default.

You can specify that more than one value is required in an attribute for successful encoding, by including the attribute:

```
MULTIPLE-VALUES
```

in the definition.

You may also specify either, or both, the minimum and maximum number of values permitted in an attribute, by including in the definition:

```
MULTIPLE-VALUES MINIMUM-NUMBER integer
                MAXIMUM-NUMBER integer
```

where `integer` is an integer in the range 1 through 32767.

A member containing an attribute with too few and/or too many values will then not encode successfully.

Example

```
ATTRIBUTE-TYPE
DECIMAL-NUMBER
IDENTIFIED-BY PAGE-SEQUENCE
INDEXED-BY VALUE
MINIMUM-LENGTH 1
MAXIMUM-LENGTH 10
MULTIPLE-VALUES MAXIMUM-NUMBER 2
;
```

The FREE-FORM-TEXT Attribute Type

The value of a FREE-FORM-TEXT attribute in a member containing that attribute may be a maximum of 32767 lines of text starting on the line following the keyword. Each line may contain a maximum of 248 characters. Only one FREE-FORM-TEXT attribute may appear in any member definition and, if present, it must come at the end of the definition. The text in the attribute may contain any characters, but if a terminator is detected in the first character position of a line, it is assumed that the end of the member definition has been reached.

To define a FREE-FORM-TEXT attribute, enter this member definition:

```
ATTRIBUTE-TYPE
FREE-FORM-TEXT
IDENTIFIED-BY user-keyword
[optional-clauses]
;
```

where:

> *user-keyword* is the name by which the attribute is referred to in definitions of members containing that attribute. The keyword must obey the rules for undelimited member names and must not begin with a numeral. It must not be the same as the keyword identifying any other attribute in the UDS table.
>
> *optional-clauses* may be any of the common clauses and/or any of the clauses described below.

To specify that the attribute is to be indexed, you may optionally include in the definition:

```
INDEXED-BY PRESENCE
```

To limit the number of lines that may appear in the attribute, you may optionally include in the definition of the attribute:

```
MINIMUM-LINES n
```

and/or:

```
MAXIMUM-LINES n
```

where *n* may be any integer not greater than 32767.

You may also specify the minimum and/or maximum number of characters permitted in each line, by including in the definition:

```
MINIMUM-LENGTH len
```

and/or:

```
MAXIMUM-LENGTH len
```

where `len` is any integer less than 247.

Example

```
ATTRIBUTE-TYPE
FREE-FORM-TEXT
IDENTIFIED-BY EXPLANATION-LINE
INDEXED-BY PRESENCE
MINIMUM-LINES 2
MAXIMUM-LINES 10
MAXIMUM-LENGTH 72
;
```

The TIME Attribute Type

Defining a TIME Attribute

The value of a TIME attribute in a member containing that attribute must be a time, for example `12:11:52`. The time must be in the format defined in the DCUST installation macro.

To define a TIME attribute, enter this member definition:

```
ATTRIBUTE-TYPE
TIME
IDENTIFIED-BY user-keyword
[optional-clauses]
;
```

where:

`user-keyword` is the name by which the attribute is referred to in definitions of members containing that attribute. The keyword must obey the rules for undelimited member names and must not begin with a numeral. It must not be the same as the keyword identifying any other attribute in the UDS table.

`optional-clauses` may be any of the common clauses and/or any of the clauses described below.

To specify that the attribute is to be indexed by presence or value, you may optionally include in the definition:

```
INDEXED-BY PRESENCE
```

or:

```
INDEXED-BY VALUE
```

If indexed by value, an encoded member having this attribute is indexed by the normalized value (see below) irrespective of the format in which the value is entered.

To specify the permitted values of the attribute, you may optionally include in the definition:

```
VALUES time
```

A member containing an attribute whose value is not one of the permitted values will not then encode successfully. time must be in the format in use at your installation. You can specify any number of times. If you do not know what that format is, you can alternatively include in the definition:

```
NORMALIZED-VALUES hhmmss
```

where `hh` is the hour, `mm` is the number of minutes, and `ss` is the number of seconds in the range 000000 to 235959.

Instead of specifying predefined values for the attribute, you may alternatively specify the maximum and/or minimum value of the attribute. To do this, include in the definition:

```
MINIMUM-VALUE time
MAXIMUM-VALUE time
```

where `time` must be in the format in use at your installation.

You can alternatively specify:

```
NORMALIZED-MINIMUM-VALUE hhmmss
NORMALIZED-MAXIMUM-VALUE hhmmss
```

where `hhmmss` is as described above.

You can specify that more than one value is required in an attribute for successful encoding, by including the attribute:

```
MULTIPLE-VALUES
```

in the definition.

You may also specify either, or both, the minimum and maximum number of values permitted in an attribute, by including in the definition:

```
MULTIPLE-VALUES MINIMUM-NUMBER integer
                MAXIMUM-NUMBER integer
```

where `integer` is an integer in the range 1 through 32767.

A member containing all attribute with too few and/or too many values will then not encode successfully.

Example

```
ATTRIBUTE-TYPE
TIME
IDENTIFIED-BY DAILY-HRS
INDEXED-BY VALUE
NORMALIZED-MINIMUM-VALUE 070000
NORMALIZED-MAXIMUM-VALUE 195959
;
```

Permitted Format for a TIME Value

If the DCUST installation macro has been tailored, the time must be in the format defined in the macro. Please consult your Systems Administrator or repository Controller if you are not sure what the format is.

Each element of the time must be separated by any printable character or space. If the separator is a space, comma, semi-colon, or right round bracket, the time must be enclosed in delimiters. Otherwise delimiters are optional. If the macro was not tailored, this is the format of the time:

`hour minute second`

where:

`hour` may be one or two numeric digits in the range 0 to 24.

`minute` may be two numeric digits in the range 00 to 59.

`second` may be two numeric digits in the range Go to 60. The inclusion of seconds is optional.

The KEYWORD Attribute Type

The value of a KEYWORD attribute in a member definition must be one of the values defined for that attribute in the UDS table.

There are two types of KEYWORD attribute: identified and unidentified.

Identified KEYWORD attribute

To define an identified KEYWORD attribute, enter this member definition:

```
ATTRIBUTE-TYPE
KEYWORD
IDENTIFIED-BY user-keyword
VALUES value
[optional-clauses]
;
```

where:

user-keyword is the name by which the attribute is referred to in definitions of members containing that attribute. The keyword must obey the rules for undelimited member names and must not begin with a numeral. It must not be the same as the keyword identifying ally other attribute in the UDS table.

optional-clauses may be any of the common clauses and/or the INDEXED-BY clause described below.

value must conform to the rules for undelimited member names. The maximum number of values that can be specified is 255.

To specify that the attribute is to be indexed by presence or value, you may optionally include in the definition:

```
INDEXED-BY PRESENCE
```

or

```
INDEXED-BY VALUE
```

You can specify that more than one value is required in an attribute for successful encoding, by including this attribute:

```
MULTIPLE-VALUES
```

in the definition.

You may also specify either, or both, the minimum and maximum number of values permitted in an attribute, by including in the definition:

```
MULTIPLE-VALUES MINIMUM-NUMBER integer
                MAXIMUM-NUMBER integer
```

where `integer` is an integer in the range 1 through 32767.

A member containing an attribute with too few and/or too many values will then not encode successfully.

Example

```
ATTRIBUTE-TYPE
KEYWORD
IDENTIFIED-BY
DATASET-DISPOSITION
VALUES NEW,RWD,OLD,SHR,KEEP,DELETE,PASS,CATLG,UNCTLG
MULTIPLE-VALUES MINIMUM 2
;
```

Unidentified KEYWORD attribute

To define an unidentified KEYWORD attribute, enter this member definition:

```
ATTRIBUTE-TYPE
KEYWORD
NAMED user-keyword
VALUES value
[optional clauses]
;
```

where:

> *`user-keyword`* is the name by which the attribute is referred to in interrogations. When a member containing an unidentified KEYWORD attribute is defined, only the value of the keyword is included in any member definition that includes that attribute. No identifier is required. For example, assume the UDS table contained this member definition:
>
> ```
> ATTRIBUTE-TYPE
> KEYWORD
> NAMED SECURITY-STATUS
> VALUES CONFIDENTIAL,PRIVATE,FREE-ACCESS
> ;
> ```
>
> When defining a member containing that attribute, you would not include the name SECURITY-STATUS, but would simply enter one of its values, for example PRIVATE.

`value` must conform to the rules for undelimited member names and must not begin with a numeral. The maximum number of values that can be specified is 255.

`optional-clauses` are the same as for an identified keyword.

The INTEGER Attribute Type

The value of an INTEGER attribute in a member containing that attribute may be a maximum of 31 digits, optionally preceded by a sign.

To define an INTEGER attribute, enter this member definition:

```
ATTRIBUTE-TYPE
INTEGER
IDENTIFIED-BY user-keyword
[optional-clauses]
;
```

where:

`user-keyword` is the name by which the attribute is referred to in definitions of members containing that attribute. The keyword must obey the rules for undelimited member names and must not begin with a numeral. It must not be the same as the keyword identifying any other attribute in the UDS table.

`optional-clauses` may be any of the common clauses and/or any of the clauses described below.

To specify that the attribute is to be indexed, you may optionally include in the definition:

```
INDEXED-BY PRESENCE
```

Or

```
INDEXED-BY VALUE
```

The attribute cannot be indexed by value if its maximum length is greater than 18 digits. If the attribute is indexed by value, it is indexed with leading zeros to 18 digits and a sign. For example, 125 is indexed as + 000000000000000125. This is to ensure that interrogations such as:

```
WHICH MEMBERS HAVE attribute EQUALS 125;
```

give the correct numeric value results (so that, for example, 125, 00125 and +125 are not treated as different numbers).

To specify the permitted values of the attribute, you may optionally include in the definition:

```
VALUES integer
```

A member containing an attribute whose value is not one of the permitted values will not encode successfully. integer may be up to 31 numeric digits optionally preceded by a sign character. You can specify any number of values.

Instead of specifying predefined values for the attribute, you may alternatively specify the maximum and/or minimum value of the attribute. To do this, include in the definition:

```
MINIMUM-VALUE integer
MAXIMUM-VALUE integer
```

You can specify that more than one value is required in an attribute for successful encoding, by including the attribute:

```
MULTIPLE-VALUES
```

in the definition.

You may also specify either, or both, the minimum and maximum number of values permitted in an attribute, by including in the definition:

```
MULTIPLE-VALUES MINIMUM-NUMBER integer
                MAXIMUM-NUMBER integer
```

where `integer` is an integer in the range 1 through 32767.

A member containing an attribute with too few and/or too many values will then not encode successfully.

To limit the number of digits that may appear in the attribute value, you may optionally include in the definition of the attribute:

```
MINIMUM-LENGTH n
```

and/or

```
MAXIMUM-LENGTH n
```

where *n* may be any integer not greater than 31. If no maximum length is specified, then a length of 18 is taken as the default.

Example

```
ATTRIBUTE-TYPE
INTEGER
IDENTIFIED-BY PRINT-RUN
MINIMUM-LENGTH 1
MAXIMUM-LENGTH 5
;
```

The NAME Attribute Type

The value of a NAME attribute in a member containing that attribute must obey the rules for data names.

To define a NAME attribute, enter this member definition:

```
ATTRIBUTE-TYPE
NAME
IDENTIFIED-BY user-keyword
[optional-clauses]
;
```

where:

> *`user-keyword`* is the name by which the attribute is referred to in definitions of members containing that attribute. The keyword must obey the rules for undelimited member names and must not begin with a numeral. It must not be the same as the keyword identifying any other attribute in the UDS table.
>
> *`optional-clauses`* may be any of the common clauses and/or any of the clauses described below.

To specify that the attribute is to be indexed by presence or value, you may optionally include in the definition:

```
INDEXED-BY PRESENCE
```

or

```
INDEXED-BY VALUE
```

To specify the permitted values of the attribute, you may optionally include in the definition:

```
VALUES name
```

where *`name`* must obey the rules for data names. You can specify any number of names.

Instead of specifying predefined values for the attribute, you may alternatively specify the maximum and/or minimum value of the attribute. To do this, include in the definition:

```
MINIMUM-VALUE name
MAXIMUM-VALUE name
```

A member containing an attribute whose value is not one of the permitted values will not encode successfully. name must obey the rules for data names. Whether a name comes within the maximum and minimum value is determined in accordance with the collating sequence.

To limit the number of characters that may appear in the attribute value, you may optionally include in the definition of the attribute:

```
MINIMUM-LENGTH n
```

and/or

```
MAXIMUM-LENGTH n
```

where *n* may be any integer not greater than 32.

You can specify that more than one value is required in an attribute for successful encoding, by including the attribute:

```
MULTIPLE-VALUES
```

in the definition.

You may also specify either, or both, the minimum and maximum number of values permitted in an attribute, by including in the definition:

```
MULTIPLE-VALUES MINIMUM-NUMBER integer
                MAXIMUM-NUMBER integer
```

where `integer` is an integer in the range 1 through 32767.

A member containing an attribute with too few and/or too many values will then not encode successfully.

Example

```
ATTRIBUTE-TYPE
NAME
IDENTIFIED-BY NOMENCLATURE
MINIMUM-LENGTH 2
MAXIMUM-LENGTH 32
MULTIPLE-VALUES MAXIMUM-NUMBER 4
;
```

The CHARACTER-STRING Attribute Type

The value of a CHARACTER-STRING attribute in a member containing that attribute may be a delimited string of a maximum of 256 characters. If the attribute value consists of two or more delimited strings, the strings are automatically concatenated when a member containing the attribute is encoded.

To define a CHARACTER-STRING attribute, enter this member definition:

```
ATTRIBUTE-TYPE
CHARACTER-STRING
IDENTIFIED-BY user-keyword
[optional-clauses]
;
```

where:

> `user-keyword` is the name by which the attribute is referred to in definitions of members containing that attribute. The keyword must obey the rules for undelimited member names and must not begin with a numeral. It must not be the same as the keyword identifying any other attribute in the UDS table.
>
> `optional-clauses` may be any of the common clauses and/or any of the clauses described below.

To specify that the attribute is to be indexed by presence or value, you may optionally include in the definition:

```
INDEXED-BY PRESENCE
```

Or

```
INDEXED-BY VALUE
```

If the character string plus the keyword is more than 78 characters, the attribute is indexed by presence only.

To specify the permitted values of the attribute, you may optionally include in the definition:

```
VALUES "string"
```

A member containing an attribute whose value is not one of the permitted values will not encode successfully. string may be any string of not more than 256 characters. You can specify any number of strings.

You can specify that more than one value is required in an attribute for successful encoding, by including the attribute:

```
MULTIPLE-VALUES
```

in the definition.

You may also specify either, or both the minimum and maximum number of values permitted in an attribute, by including in the definition:

```
MULTIPLE-VALUES MINIMUM-NUMBER integer
                MAXIMUM-NUMBER integer
```

where `integer` is an integer in the range 1 through 32767.

A member containing an attribute with too few and/or too many values will then not encode successfully.

Instead of specifying predefined values for the attribute, you may alternatively specify the maximum and/or minimum value of the attribute. To do this, include in the definition:

```
MINIMUM-VALUE "string"
MAXIMUM-VALUE "string"
```

You can also specify the maximum and/or minimum length of the attribute value. To do this, include in the definition:

```
MINIMUM-LENGTH n
MAXIMUM-LENGTH n
```

where *n* may be an integer in the range 1 through 256.

Example

```
ATTRIBUTE-TYPE
CHARACTER-STRING
IDENTIFIED-BY AUDITED
INDEXED-BY VALUE
;
```

The TEXT Attribute Type

The value of a TEXT attribute in a member containing that attribute may be a maximum of 32767 delimited character strings, each containing a maximum of 246 characters. Each string may contain any printable characters including spaces.

If a member containing this attribute is processed by a REPORT or GLOSSARY command, the strings in the attribute are not concatenated but are aligned one beneath the other on the opening delimiters, even when several strings appear on a single input line. This allows you to include, for example, tables and lists in a member's documentation in the repository.

To define a TEXT attribute, enter the following member definition:

```
ATTRIBUTE-TYPE
TEXT
IDENTIFIED-BY user-keyword
[optional-clauses]
;
```

where:

> *user-keyword* is the name by which the attribute is referred to in definitions of members containing that attribute. The keyword must obey the rules for undelimited member names and must not begin with a numeral. It must not be the same as the keyword identifying any other attribute in the UDS table.
>
> *optional-clauses* may be any of the common clauses and/or any of the clauses described below.

To specify that the attribute is to be indexed, you may optionally include in the definition:

```
INDEXED-BY PRESENCE
```

This attribute may not be indexed by value.

To limit the number of lines that may appear in the attribute value, you may optionally include in the definition of the attribute:

```
MINIMUM-LINES n
```

and/or

```
MAXIMUM-LINES n
```

where *n* may be any integer not greater than 32767.

You may also specify the minimum and/or maximum number of characters permitted in each line, by including in the definition:

```
MINIMUM-LENGTH len
```

and/or

```
MAXIMUM-LENGTH len
```

where `len` is any integer less than 247.

Example

```
ATTRIBUTE-TYPE
TEXT
IDENTIFIED-BY CIRCULATION
INDEXED-BY PRESENCE
MAXIMUM-LENGTH 100
;
```

Indexing User Defined Attributes

A user-defined attribute may be indexed by including the INDEXED-BY clause in the definition of the attribute. This is the syntax:

```
INDEXED-BY PRESENCE
```

Or

```
INDEXED-BY VALUE
```

Some attributes may be indexed by presence alone.

When a member containing all indexed attribute is encoded, the presence of the attribute is recorded in the Index data set of the repository. This extends the ways in which the attribute can be processed as follows:

- The attribute will appear in the output from the LIST INDEX-NAMES command or from the LIST ATTRIBUTES command (subject to any other selection criteria in the command).
- The attribute can be selected by the PERFORM and WHICH commands.
- The attribute can be identified by the WHAT IS command.

In addition, the Index dataset is designed to achieve the fastest possible retrieval of names recorded in the data set. The value of any attribute, whether indexed or not, can be Output by the GLOSSARY command or interrogated by the WHICH command. However indexed attributes can be processed faster than attributes that are not indexed.

If the attribute is indexed by presence alone, a single entry is created on the Index data set containing a pointer to each member with that attribute.

If the attribute is indexed by value, an entry is created containing pointers to the members with that attribute. One or more other entries are created, one for each different value of the attribute, containing pointers to the members containing the attribute with that value.

For example, assume that you have an attribute named COLOR which is indexed by value. The attribute may have the values BLUE, RED, or GREEN. When the first member containing the attribute:

```
COLOR RED
```

is encoded, an entry for the attribute COLOR is created on the index dataset and another for the attribute value COLOR = RED, each containing a pointer to that member. If a member containing the attribute:

```
COLOR GREEN
```

is then encoded, the entry for the attribute COLOR is updated to contain a pointer to that member and a new entry is created for the attribute value COLOR GREEN. If a further member containing the attribute:

```
COLOR RED
```

is encoded, the entries for the attribute COLOR and for the attribute value COLOR = RED are both updated to contain pointers to that member.

6 Defining an Attribute Group

An ATTRIBUTE-GROUP groups ATTRIBUTE-TYPEs and/or ATTRIBUTE-GROUPs together so that they can be collectively referred to. This is the syntax of an ATTRIBUTE-GROUP:

```
ATTRIBUTE-GROUP
CONTAINS member-name [, member-name]...
⎧ : ⎫
⎩ ; ⎭
```

where *`member-name`* is the name of an ATTRIBUTE-TYPE or ATTRIBUTE-GROUP. If the definition refers to an ATTRIBUTE-GROUP then all the attributes in that group are included in the ATTRIBUTE-GROUP being defined.

Each name specified may be followed by an OPTIONAL sub-clause which states whether or not that attribute needs to be present when members whose MEMBER-TYPE definitions include the ATTRIBUTE-GROUP are encoded.

Clause	**Effect**
OPTIONAL YES	The attribute need not be present.
OPTIONAL NO	The attribute need not be present.
OPTIONAL WARN	The attribute must be present.

The default is OPTIONAL YES.

The attribute need not be present, but a warning message is issued if it is not.

You can specify that ATTRIBUTE-TYPEs (but not ATTRIBUTE-GROUPs) are mutually exclusive by including an ELSE sub-clause, thus:

```
CONTAINS attribute-type [ELSE attribute-type]...
```

The ELSE sub-clause specifies that where one ATTRIBUTE-TYPE is present, the other must not be. For example:

```
CONTAINS EGBFRFM ELSE EGBKEY
```

indicates that members in the hierarchy may contain either the attribute defined by EGBFRFM or the attribute defined by EGBKEY, but not both. If two or more ELSE clauses are included. all ATTRIBUTE-TYPEs specified are mutually exclusive.

For any set of mutually exclusive ATTRIBUTE-TYPEs that you specify, an OPTIONAL sub-clause may follow the last ELSE sub-clause. OPTIONAL sub-clauses may not be included for individual ATTRIBUTE-TYPEs named within the set.

```
ATTRIBUTE-GROUP
CONTAINS attribute-clause [, attribute-clause]...
[common-clauses]
{ . }
{ ; }
```

where:

attribute-clause is:

```
 { attribute-type [ ELSE attribute-type ]...  }
 { attribute-group                            }
   [OPTIONAL  { YES  } ]
              { NO   }
              { WARN }
```

common-clauses are as defined in Chapter 9, "Common Clauses," on page 63.

7 Interrogating the Contents of a UDS Table

Use the SHOW UDS command to interrogate the contents of a UDS table.

Refer to "Interrogating the UDS Table" on page 3 for an overview of the SHOW UDS command.

Refer to the *ASG-Manager Products Dictionary/Repository User's Guide* for full details of the SHOW UDS command.

8 Looking Ahead: Planning for Future Attributes

If at some future time you want to make new attributes available for the members in your dictionary, you will need a new UDS table which includes those attributes. The new UDS table will be constructed onto the MP-AID by the Controller.

Before applying the new UDS table to the dictionary, the Controller will issue a command that compares the new UDS table with the old one, to assess their compatibility. The output from this command may indicate a need to BULK ENCODE all members of the type(s) to which the new attributes can apply. This can arise because, in some circumstances, making new attributes available for inclusion in a member type results in a need to reserve an additional byte in the encoded records of members of that type. The technique described below avoids the need to reserve this additional byte; and hence avoids the possibility that you may need to re-encode all your members if in the future new attributes are required for them.

If a member being encoded contains a reference to a member name for which no data entries record exists in the dictionary, then a dummy member is created with that name. The member type for the dummy member depends on the referring member's member type and the clause in which the reference appears. If the reference appears in one of these:

- The COMMON-ATTRIBUTES clause of a HIERARCHY member, or
- The ATTRIBUTES clause of a MEMBER-TYPE member, or
- The GENERIC-ATTRIBUTES clause of a MEMBER-TYPE member, or
- The CONTAINS clause of an ATTRIBUTE-GROUP member

then a dummy ATTRIBUTE-GROUP is created.

A UDS table may be constructed from a hierarchy which includes dummy ATTRIBUTE-GROUP members. If one of these ATTRIBUTE-GROUPs is subsequently defined, and a new UDS table is constructed (to include the new attributes contained in that group) and applied to the dictionary, it will not be necessary to reserve additional space in the encoded records of members of the affected type(s). The possible presence of the attribute group would already have been allowed for; so it will not be necessary to reencode members until they are actually updated to include the new attributes.

If you foresee a possibility that further attributes may be required, therefore, you can avoid the need to bulk re-encode your members at that time by including a reference to a dummy member in each of the above-listed clauses that you define for your hierarchy. Then when additional attributes are required, you can define for the appropriate dummy an ATTRIBUTE-GROUP member that contains the new attributes. This member should also contain a reference to a new dummy ATTRIBUTE-GROUP, to allow for further developments.

9 Common Clauses

Common clauses are fully defined in the *ASG-Manager Products Dictionary/Repository User’s Guide*. However, for ease of reference, their syntax is repeated below.

```
ACCESS-AUTHORITY 'name' { READ-ONLY        }
                        { UPDATE           }
                        { SECURITY-CONTROL }

        [,'name' { READ-ONLY        } ]...
                 { UPDATE           }
                 { SECURITY-CONTROL }

ADMINISTRATIVE-DATA 'text'

ALIAS [alias-type] 'alias' [, [alias-type] 'alias']...

CATALOGUE 'classification' [, 'classification'}...

COMMENT 'text'

DESCRIPTION 'text'

EFFECTIVE-DATE date

FREQUENCY { frequency                                                     }
          { { ACCESS } frequency [ { ACCESS } frequency ]...              }
          { { RUN    }             { RUN    }                             }
          { { UPDATE }             { UPDATE }                             }
          { { BACK-UP}             { BACK-UP}                             }

NOTE 'text'

OBSOLETE-DATA date

QUERY 'text'

SECURITY-CLASSIFICATION 'string' [FROM date]
                   ['string' [FROM date]]...

SEE member-name [FOR 'string']
                   [, member-name [FOR 'string']]...
```

where `frequency` is:

```
{ MONTHLY } { n        }
{ WEEKLY  } { 'string' }
{ DAILY   }
{ HOURLY  }
```

10 UDS Load Modules Supplied by ASG

This is a list of UDS load modules supplied by ASG and the Manager Products that they support. These modules constitute a preferred set of UDS tables that you can use for your dictionaries. If a UDS table cannot be found on the MP-AID when the dictionary controller is comparing UDS tables or applying a UDS table to a dictionary, then ControlManager looks for it in the library.

UDS Load Module	ASG-Manager Product Supported
DU001	DataManager, Designmanager, DictionaryManager, SourceManager, and ControlManager.
DU002	DataManager, DesignManager, SourceManager, and the English language version of the Extended Data Processing Structure (EDPS).
DU003	DataManager, DesignManager, SourceManager, and the Structured Analysis Structure (SAS).
DU004	DataManager, DesignManager, SourceManager, EDPS, and SAS.
DU006	DataManager, DesignManager, SourceManager, and the Structured Development Structure (SDS).
DU007	ControlManager, DictionaryManager, and UDS.
DU012	DataManager, DesignManager, DictionaryManager, SourceManager, ControlManager, UDS, and EDPS.
DUE02	DataManager, DesignManager, SourceManager, and the Spanish language version of EDPS.
DUF02	DataManager, DesignManager, SourceManager, and the French language version of EDPS.
DUG02	DataManager, DesignManager, SourceManager, and the German language version of EDPS.
DUH02	DataManager, DesignManager, SourceManager, and the Dutch language version of EDPS.
DUI02	DataManager, DesignManager, SourceManager, and the Italian language version of EDPS.

UDS Load Module	ASG-Manager Product Supported
DUK02	DataManager, DesignManager, SourceManager, and the Danish language version of EDPS.
DUN02	DataManager, DesignManager, SourceManager, and the Norwegian language version of EDPS.
DUS02	DataManager, DesignManager, SourceManager, and the Swedish language version of EDPS.
DUU02	DataManager, DesignManager, SourceManager, and the Finnish language version of EDPS.

Note:

Where support for ControlManager is shown, that support excludes the UDS member types provided by selectable unit CMR-UD1. Support for the UDS member types is shown separately.

DU001 is used as a default UDS table when a dictionary is first created.

DU007 is intended for the Manager Products Administration Dictionary in UDS environments.

Appendix A

Relationships Permitted from Base Member Types

By default, a relationship clause in a user-defined member may refer to members of any of the types to which the clause can refer in the base member type, or based on any of such types.

The sections in this chapter list:

- The clauses in the base member types that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in these tables are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Base Member Types

The base member types to which the tables given in these panels relate are listed below. Note that DataManager IMS (DL/I) Interface members are not included because they cannot be used as base member types.

ControlManager Member Types

- ATTRIBUTE-GROUP
- ATTRIBUTE-TYPE
- EXECUTIVE-ROUTINE
- FORMAT
- GLOBAL-PROFILE
- HIERARCHY
- INFOBANK-PANEL
- LOGON-PROFILE
- MEMBER-TYPE
- MEMBER-TYPE-GROUP
- OBSOLETE-DEFINITION
-

DataManager Member Types

- ADABAS-DATABASE
- ADABAS-FILE
- COMMAND-STREAM
- FILE
- GROUP
- IDMS-AREA
- IDMS-DATABASE
- IDMS-LOGICAL-RECORD
- IDMS-PATH-GROUP
- IDMS-RECORD
- IDMS-SET
- IDMS-SUBSCHEMA
- IDMS-VIEW
- ITEM
- MARKIV-FILE
- MARKIV-SEGMENT
- MODULE
- PROGRAM
- SESAM-STORAGE
- SESAM-TABLE
- SESAM-VIEW
- SUDS-AREA
- SUDS-DATABASE
- SUDS-RECORD
- SUDS-SET
- SUDS-SUBSCHEMA
- SUDS-VIEW
- SYSTEM
- SYSTEM-2000-DATABASE
- SYSTEM-2000-SCHEMA-RECORD
- SYSTEM-2000-SUBSCHEMA-RECORD
- TOTAL-DATABASE
- TOTAL-MASTER-FILE
- TOTAL-VARIABLE-FILE

DesignManager Member Types

- ENTITY
- FORMAT
- GROUP
- ITEM
- USERVIEW
- VIEWSET

DictionaryManager Member Type

- TRANSLATION-RULE

SourceManager Member Types

- DL/ 1-AREA
- REUSABLE-CODE

Relationships Permitted from ATTRIBUTE-GROUP Member Types

This table lists:

- The clauses in the ATTRIBUTE-GROUP member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
CONTAINS	ATTRIBUTE-TYPE ATTRIBUTE-GROUP	ATTRIBUTE-TYPE
SEE	Any member type	ATTRIBUTE-GROUP

Relationships Permitted from ATTRIBUTE-TYPE Member Types

This table lists:

- The clauses in the ATTRIBUTE-TYPE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
SEE	Any member type	ATTRIBUTE-TYPE

Relationships Permitted from EXECUTIVE-ROUTINE Member Types

This table lists:

- The clauses in the EXECUTIVE-ROUTINE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
SEE	Any member type	EXECUTIVE-ROUTINE

Relationships Permitted from FORMAT Member Types

This table lists:

- The clauses in the FORMAT member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
SEE	Any member type	FORMAT

Relationships Permitted from GLOBAL-PROFILE Member Types

This table lists:

- The clauses in the GLOBAL-PROFILE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
SEE	Any member type	GLOBAL-PROFILE

Relationships Permitted from HIERARCHY Member Types

This table lists:

- The clauses in the HIERARCHY member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
COLLECTIVE	MEMBER-TYPE MEMBER-TYPE-GROUP	MEMBER-TYPE
COMMON-ATTRIBUTES	ATTRIBUTE-TYPE ATTRIBUTE-GROUP	ATTRIBUTE-TYPE
CONTAINS	MEMBER-TYPE MEMBER-TYPE-GROUP	MEMBER-TYPE
SEE	Any member typeq	HIERARCHY
SYNONYM	MEMBER-TYPE MEMBER-TYPE-GROUP	MEMBER-TYPE
UDO	ATTRIBUTE-TYPE	ATTRIBUTE-TYPE

Relationships Permitted from INFOBANK-PANELS Member Types

This table lists:

- The clauses in the INFOBANK-PANEL member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
LAST-MENU	INFOBANK-PANEL	INFOBANK-PANEL
LAST-PANEL	INFOBANK-PANEL	INFOBANK-PANEL
NEXT-PANEL	INFOBANK-PANEL	INFOBANK-PANEL
SEE	Any member type	INFOBANK-PANEL
SELECT	INFOBANK-PANEL	INFOBANK-PANEL

Relationships Permitted from LOGON-PROFILE Member Types

This table lists:

- The clauses in the LOGON-PROFILE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
BATCH-GLOBAL	GLOBAL-PROFILE	GLOBAL-PROFILE
COMMON-GLOBAL	GLOBAL-PROFILE	GLOBAL-PROFILE
ONLINE-GLOBAL	GLOBAL-PROFILE	GLOBAL-PROFILE
SEE	Any member type	LOGON-PROFILE

Relationships Permitted from MEMBER-TYPE Member Types

This table lists:

- The clauses in the LOGON-PROFILE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
ATTRIBUTES	ATTRIBUTE-TYPE ATTRIBUTE-GROUP	ATTRIBUTE-TYPE
BASED-ON	MEMBER-TYPE	MEMBER-TYPE
GENERIC- ATTRIBUTES	ATTRIBUTE-TYPE ATTRIBUTE-GROUP	ATTRIBUTE-TYPE
IS	MEMBER-TYPE	MEMBER-TYPE
RELATIONSHIPS	MEMBER-TYPE MEMBER-TYPE-GROUP	MEMBER-TYPE
SEE	Any member type	MEMBER-TYPE

Relationships Permitted from MEMBER-TYPE-GROUP Member Types

This table lists:

- The clauses in the MEMBER-TYPE-GROUP member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
CONTAINS	MEMBER-TYPE MEMBER-TYPE-GROUP	MEMBER-TYPE
SEE	Any member type	MEMBER-TYPE

Relationships Permitted from OBSOLETE-DEFINITION Member Types

This table lists:

- The clauses in the OBSOLETE-DEFINITION member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
SEE	Any member type	OBSOLETE-DEFINITION

Relationships Permitted from ADABAS-DATABASE Member Types

This table lists:

- The clauses in the ADABAS-DATABASE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
CIPHER	ADABAS-FILE	ADABAS-FILE
CIPHER-BY	ITEM	ITEM
CONTAINS	ADABAS-FILE	ADABAS-FILE
COUPLE	ADABAS-FILE	ADABAS-FILE
COUPLE-BY	ITEM GROUP	ITEM
SEE	Any member type	ADABAS-DATABASE

Relationships Permitted from ADABAS-FILE Member Types

This table lists:

- The clauses in the ADABAS-FILE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
CONTAINS	ITEM and GROUP	ITEM
BOUND	ITEM	ITEM
DESCRIPTORS	ITEM and GROUP	ITEM
SUB-DESCRIPTORS	ITEM and GROUP	ITEM
SUPER-DESCRIPTORS	ITEM and GROUP	ITEM
FILED-NAMES	ITEM and GROUP	ITEM
PHONETIC-NAMES	ITEM and GROUP	ITEM
SEE	Any member type	ADABAS-FILE

Relationships Permitted from COMMAND-STREAM Member Types

This table lists:

- The clauses in the COMMAND-STREAM member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
SEE	Any member type	COMMAND-STREAM

Relationships Permitted from FILE Member Types

This table lists:

- The clauses in the FILE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
BOUND	ITEM	ITEM
CONTAINS	ITEM FILE GROUP	ITEM
IF	ITEM	ITEM
SEE	Any member type	FILE
SORT-KEYS	ITEM GROUP	ITEM
USER-LABELS	MODULE	MODULE

Relationships Permitted from GROUP Member Types

This table lists:

- The clauses in the GROUP member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
BOUND	ITEM	ITEM
CONTAINS	ITEM GROUP	ITEM
IF	ITEM	ITEM
KEYS	ITEM GROUP	ITEM
SEE	Any member type	GROUP
USER-EXIT	MODULE REUSABLE-CODE	MODULE

Relationships Permitted from IDMS-AREA Member Types

This table lists:

- The clauses in the IDMS-AREA member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
CALL	MODULE PROGRAM SYSTEM	MODULE
IN	FILE	FILE
SEE	Any member type	IDMS-AREA

Relationships Permitted from IDMS-DATABASE Member Types

This table lists:

- The clauses in the IDMS-DATABASE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
AREAS	IDMS-AREA	IDMS-AREA
DATASETS	FILE	FILE
SEE	Any member type	IDMS-DATABASE
SETS	IDMS-SET	IDMS-SET
STAND-ALONE	IDMS-RECORD	IDMS-RECORD

Relationships Permitted from IDMS-LOGICAL-RECORD Member Types

This table lists:

- The clauses in the IDMS-LOGICAL-RECORD member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
CONTAINS	IDMS-RECORD	IDMS-RECORD
ERASE	IDMS-PATH	IDMS-PATH
MODIFY	IDMS-PATH	IDMS-PATH
OBTAINS	IDMS-PATH	IDMS-PATH
SEE	Any member type	IDMS-RECORD
STORE	IDMS-PATH	IDMS-PATH

Relationships Permitted from IDMS-PATH-GROUP Member Types

This table lists:

- The clauses in the IDMS-PATH-GROUP member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
COMPUTE	ITEM GROUP IDMS-SET	ITEM
CONNECT	IDMS-RECORD	IDMS-RECORD
CONNECT-TO	IDMS-SET	IDMS-SET
DISCONNECT	IDMS-RECORD	IDMS-RECORD
DISCONNECT- FROM	IDMS-SET	IDMS-SET
ERASE	IDMS-RECORD	IDMS-RECORD
EVALUATE	ITEM GROUP IDMS-SET	ITEM
FIND	IDMS-RECORD	IDMS-RECORD
FIND-USING	ITEM GROUP IDMS-SET	ITEM
FOR	ITEM GROUP IDMS-RECORD	ITEM
GET	IDMS-RECORD	IDMS-RECORD
KEEP	IDMS-RECORD	IDMS-RECORD
MODIFY	IDMS-RECORD	IDMS-RECORD
PATHS	IDMS-PATH	IDMS-PATH

Clause	Permitted member types	Dummy
SEE	Any member type	IDMS-PATH
STORE	IDMS-RECORD	IDMS-RECORD
WITHIN	GROUP IDMS-RECORD IDMS-SET IDMS-AREA	GROUP

Relationships Permitted from IDMS-RECORD Member Types

This table lists:

- The clauses in the IDMS-RECORD member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
AREA	IDMS-AREA	IDMS-AREA
BOUND	ITEM	ITEM
CALL	MODULE PROGRAM SYSTEM	MODULE
CONTAINS	ITEM GROUP	ITEM
IF	ITEM	ITEM
KEYS	ITEM GROUP	ITEM
SEE	Any member type	IDMS-RECORD
STORED-VIA	IDMS-SET	IDMS-SET

Relationships Permitted from IDMS-SET Member Types

This table lists:

- The clauses in the IDMS-SET member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
MEMBER	IDMS-RECORD	IDMS-RECORD
OWNER	IDMS-RECORD	IDMS-RECORD
SEE	Any member type	IDMS-SET
SORTED	ITEM GROUP	ITEM

Relationships Permitted from IDMS-SUBSCHEMA Member Types

This table lists:

- The clauses in the IDMS-SUBSCHEMA member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	**Permitted member types**	**Dummy**
ACCESSES	IDMS-DATABASE	WMS-DATABASE
AREA	IDMS-AREA	IDMS-AREA
DMCL-AREA	IDMS-AREA	IDMS-AREA
DMCL-AS	IDMS-SUBSCHEMA	IDMS-SUBSCHEMA
JOURNAL	FILE	FILE
LOGICAL-RECORD	IDMS-LOGICAL-RECORD	IDMS-LOGICAL-RECORD
RECORD	IDMS-RECORD	IDMS-RECORD
SEE	Any member type	IDMS-SUBSCHEMA
SELECTING	ITEM GROUP	ITEM
SET	IDMS-SET	IDMS-SET
STATISTICS-FOR	PROGRAM	PROGRAM
STATISTICS-OF	IDMS-DATABASE	IDMS-DATABASE
STATISTICS-TO	IDMS-SUBSCHEMA	IDMS-SUBSCIIEMA
VIEW	IDMS-VIEW	IDMS-VIEW

Relationships Permitted from IDMS-VIEW Member Types

This table lists:

- The clauses in the IDMS-VIEW member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
RECORD	IDMS-RECORD	IDMS-RECORD
SEE	Any member type	IDMS-VIEW
SELECTING	ITEM GROUP	ITEM

Relationships Permitted from ITEM Member Types

This table lists:

- The clauses in the ITEM member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
IF	ITEM	ITEM
NAME	ITEM	ITEM
SEE	Any member type	ITEM
USER-EXIT	MODULE REUSABLE-CODE	MODULE

Relationships Permitted from MARKIV-FILE Member Types

This table lists:

- The clauses in the MARKIV-FILE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permltted member types	Dummy
CONTAINS	MARKIV-SEGMENT ITEM GROUP	MARKIV-SEGMENT
COUNT-FIELD	ITEM	ITEM
PARENT	MARKIV-SEGMENT ITEM GROUP	MARKIV-SEGMENT
SEE	Any member type	MARKIV-FILE
SORT-KEYS	ITEM GROUP	ITEM
USER-LABELS	MODULE	MODULE

Relationships Permitted From MARKIV-SEGMENT Member Types

This table lists:

- The clauses in the MARKIV-SEGMENT member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
BOUND	ITEM	ITEM
CONTAINS	ITEM GROUP	ITEM
IF	ITEM	ITEM
KEYS	ITEM GROUP	ITEM
SEE	Any member type	MARKIV-SEGMENT
USER-EXIT	MODULE	MODULE

Relationships Permitted From MODULE Member Types

This table lists:

- The clauses in the MODULE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type..

Clause	Permitted member types	Dummy
CALLS	MODULE	MODULE
CONTAINS	Any DATA member type MODULE	ITEM
INPUTS	Any DATA member type	ITEM
OUTPUTS	Any DATA member type	ITEM
PARAMETERS	Any DATA member type	ITEM
PASSING	Any DATA member type	ITEM
UPDATES	Any DATA member type	ITEM

These clauses are available only if the relevant database management system interface facility is installed.

Clause	Permitted member types	Dummy
ACCESS	MODULE PROGRAM SYSTEM	MODULE
ACCESSES	SYSTEM-2000-DATABASE	SYSTEM-2000-DATABASE
ACCESS-FILE	ADABAS-FILE	ADABAS-FILE
COMMBLOCK	SYSTEM-2000-DATABASE	SYSTEM-2000-DATABASE
COMMBLOCK-MEMBER	GROUP	GROUP
COUNTS-AS	ITEM	ITEM
EDIT-NAME	ITEM	ITEM
GIVING	ITEM GROUP	ITEM
GIVING-(THROUGH)	ITEM GROUP	ITEM
GIVING-THROUGH	ITEM GROUP	ITEM
GIVING-IN	GROUP	GROUP
PROCESSES-IDMS	IDMS-SUBSCHEMA	IDMS-SUBSCHEMA
PROCESSES-IMS-CONTAINS	STRUCTURE-PCB GSAM-PCB OUTPUT-MESSAGE-PCB	STRUCTURE-PCB
PROCESSES-SUDS	SUDS-SUBSCHEMA	SUDS-SUBSCHEMA
QUALIFIED-ON	ITEM GROUP DLI -SEQU-KEY-PHYSICAL-SEG DLI-SEQU-KEY-INDEX-PTR-SEG INDEX-SEARCH-FIELD SYSTEM-RELATED-FIELD/CK SYSTEM -RELATED-FIELD/SX CONCATENATED-KEY	ITEM
SEE	Any member type	SYSTEM
SELECTING	ITEM GROUP	ITEM
SOURCE-SSR	SYSTEM-2000-SCHEMA-RECORD GROUP S2K-SUBSCHEMA-RECORD	SYSTEM-2000-SCHEMA-RECORD

Clause	Permitted member types	Dummy
SSAS	PHYSICAL-SEGMENT UNIDIRECTIONAL-LOG-CHILD PHYSICALLY-PAIRED-LOG-CHI LD REAL-PAIRED-LOG-CHILD VIRTUAL-PAIRED-LOG-CHILD SOURCE-SEGMENT DEST-PARENT/SOURCE SEGMENT TARGET-SEGMENT DEST-PARENT/TARGET-SEGME NT SOURCE/TARGET-SEGMENT DEST-PARENT/SOURCE/ TARGET-SEG LOGICAL-SEGMENT LOGICAL-CONCATENATED-SEG INDEX-POINTER-SEGMENT	PHYSICAL-SEGMENT
SUBSCHEMA	S2K-SUBSCHEMA-RECORD	S2K-SUBSCHEMA-RECORD
TARGET-SSR	SYSTEM-2000-SCHEMA-RECOR D GROUP S2K-SUBSCHEMA-RECORD	SYSTEM-2000-SCHEMA-RECORD
USER-PASSWORD	ITEM	ITEM
VIEWS	SYSTEM-2000-SCHEMA-RECOR D	SYSTEM-2000-SCHEMA-RECORD

Relationships Permitted From PROGRAM Member Types

This table lists:

- The clauses in the PROGRAM member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
CALLS	MODULE PROGRAM REUSABLE-CODE	MODULE
CONTAINS	Any DATA member type MODULE PROGRAM REUSABLE-CODE	ITEM
INPUTS	Any DATA member type	ITEM
OUTPUTS	Any DATA member type	ITEM
PARAMETERS	Any DATA member type	ITEM
PASSING	Any DATA member type	ITEM
UPDATES	Any DATA member type	ITEM

These clauses are available only if the relevant database management system interface facility is installed.

Clause	Permitted member types	Dummy
ACCESS	MODULE PROGRAM SYSTEM	MODULE
ACCESSES	SYSTEM-2000-DATABASE	SYSTEM-2000-DATABASE
ACCESS-FILE	ADABAS-FILE	ADABAS-FILE
COMMBLOCK	SYSTEM-2000-DATABASE	SYSTEM-2000-DATABASE
COMMBLOCK-M EMBER	GROUP	GROUP
COUNTS-AS	ITEM	ITEM
EDIT-NAME	ITEM	ITEM
GIVING	ITEM GROUP	ITEM
GIVING- (THROUGH)	ITEM GROUP	ITEM
GIVING- THROUGH	ITEM GROUP	ITEM
GIVING-IN	GROUP	GROUP
PROCESSES- IDMS	IDMS-SUBSCHEMA	IDMS-SUBSCHEMA

Clause	Permitted member types	Dummy
PROCESSES-IMS-CONTAINS	STRUCTURE-PCB GSAM-PCB OUTPUT-MESSAGE-PCB	STRUCTURE-PCB
PROCESSES-SUDS	SUDS-SUBSCHEMA	SUDS-SUBSCHEMA
QUALIFIED-ON	ITEM GROUP DLI -SEQU-KEY-PHYSICAL-SEG DLI -SEQU-KEY-INDEX-PTR-SEG INDEX-SEARCH-FIELD SYSTEM-RELATED-FIELD/CK SYSTEM -RELATED-FIELD/SX CONCATENATED-KEY	ITEM
SEE	Any member type	SYSTEM
SELECTING	ITEM GROUP	ITEM
SOURCE-SSR	SYSTEM-2000-SCHEMA-RECORD GROUP S2K-SUBSCHEMA-RECORD	SYSTEM-2000-SCHEMA-RECORD
SSAS	PHYSICAL-SEGMENT UNIDIRECTIONAL-LOG-CHILD PHYSICALLY-PAIRED-LOG-CHILD REAL-PAIRED-LOG-CHILD VIRTUAL-PAIRED-LOG-CHILD DESTINATION-PARENT-SEG SOURCE-SEGMENT DEST-PARENT/SOURCE SEGMENT TARGET-SEGMENT DEST-PARENT/TARGET-SEGMENT SOURCE/TARGET-SEGMENT DEST-PARENT/SOURCE/ TARGET-SEG LOGICAL-SEGMENT LOGICAL-CONCATENATED-SEG INDEX-POINTER-SEGMENT	PHYSICAL-SEGMENT
SUBSCHEMA	S2K-SUBSCHEMA-RECORD	S2K-SUBSCHEMA-RECORD
TARGET-SSR	SYSTEM-2000-SCHEMA-RECORD GROUP S2K-SUBSCHEMA-RECORD	SYSTEM-2000-SCHEMA-RECORD
USER-PASSWORD	ITEM	ITEM
VIEWS	SYSTEM-2000-SCHEMA-RECORD	SYSTEM-2000-SCHEMA-RECORD

Relationships Permitted From SESAM-STORAGE Member Types

This table lists:

- The clauses in the MARKIV-SEGMENT member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
COMPOUND-KEY	ITEM	ITEM
CONTAINS	SESAM-TABLE	SESAM -TABLE
KEY	ITEM	ITEM
SECONDARY-KEYS	ITEM	ITEM
SEE	Any member type	SESAM -STORAGE

Relationships Permitted From SESAM-TABLE Member Types

This table lists:

- The clauses in the SESAM-TABLE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
CONTAINS	ITEM GROUP	ITEM
KEY-VALUES-FOR	ITEM	ITEM
SEE	Any member type	SESAM -TABLE
SYMBOLIC-NAMES-FOR	ITEM	ITEM

Relationships Permitted From SESAM-VIEW Member Types

This table lists:

- The clauses in the SESAM-VIEW member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
SEE	Any member type	SESAM-VIEW
SELECT-FROM	SESAM-TABLE	SESAM-TABLE
SELECT-MEMBERS	ITEM GROUP	GROUP

Relationships Permitted from SUDS-AREA Member Types

This table lists:

- The clauses in the SUDS-AREA member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
CONTAINS	SUDS-RECORD	SUDS-RECORD
SEE	Any member type	SUDS-AREA

Relationships Permitted from SUDS-DATABASE Member Types

This table lists:

- The clauses in the SUDS-DATABASE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
AREAS	SUDS-AREA	SUDS-AREA
SEE	Any member type	SUDS-AREA
SETS	SUDS-SET	SUDS-SET

Relationships Permitted from SUDS-RECORD Member Types

This table lists:

- The clauses in the SUDS-RECORD member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
CONTAINS	ITEM GROUP	ITEM
DATABASE-KEYS	ITEM	ITEM
IN	SUDS-RECORD	SUDS-RECORD
KEYS	ITEM	ITEM
LOCATED-IN	SUDS-AREA	SUDS-AREA
PLACED-IN	SUDS-AREA	SUDS-AREA
PLACEMENT-SET	SUDS-SET	SUDS-SET
POPULATION-FOR	SUDS-AREA	SUDS-AREA
PROCEDURE	MODULE	MODULE
SEARCH-KEYS	ITEM	ITEM
SEE	Any member type	SUDS-RECORD

Relationships Permitted from SUDS-SET Member Types

This table lists:

- The clauses in the SUDS-SET member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
IN	SUDS-RECORD	SUDS-RECORD
MEMBER	SUDS-RECORD	SUDS-RECORD
OWNER	SUDS-RECORD	SUDS-RECORD
PLACED-IN	SUDS-AREA	SUDS-AREA
PROCEDURE	MODULE	MODULE
SEARCH-KEYS	ITEM	ITEM
SEE	Any member type	SUDS-SET
SORT-KEYS	ITEM	ITEM
WITHIN	SUDS-AREA	SUDS-AREA

Relationships Permitted From SUDS-SUBSCHEMA Member Types

This table lists:

- The clauses in the SUDS-SUBSCHEMA member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
ACCESSES	SUDS-DATABASE	SUDS-DATABASE
AREA	SUDS-AREA	SUDS-AREA
RECORDS	SUDS-RECORD SUDS-VIEW	SUDS-RECORD
SEE	Any member type	SUDS-.SUBSCHEMA
SET	SUDS-SET	SUDS-SET

Relationships Permitted From SUDS-VIEW Member Types

This table lists:

- The clauses in the SUDS-VIEW member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
CONTAINS	ITEM, GROUP	ITEM
DATABASE-KEYS	ITEM	ITEM
RECORD	SUDS-RECORD	SUDS-RECORD
SEE	Any member type	SUDS-VIEW

Relationships Permitted from SYSTEM Member Types

This table lists:

- The clauses in the SYSTEM member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
CALLS	MODULE PROGRAM SYSTEM REUSABLE-CODE	MODULE
CONTAINS	Any DATA member type MODULE PROGRAM SYSTEM REUSABLE-CODE	ITEM
INPUTS	Any DATA member type	ITEM
OUTPUTS	Any DATA member type	ITEM
PARAMETERS	Any DATA member type	ITEM
PASSING	Any DATA member type	ITEM
UPDATES	Any DATA member type	ITEM

These clauses are available only if the relevant database management system interface facility is installed.

Clause	Permitted member types	Dummy
ACCESS	MODULE PROGRAM SYSTEM	MODULE
ACCESSES	SYSTEM-2000-DATABASE	SYSTEM-2000-DATABASE
ACCESS-FILE	ADABAS-FILE	ADABAS-FILE
COMMBLOCK	SYSTEM-2000-DATABASE	SYSTEM-2000-DATABASE

Clause	Permitted member types	Dummy
COMMBLOCK-MEMBER	GROUP	GROUP
COUNTS-AS	ITEM	ITEM
EDIT-NAME	ITEM	ITEM
GIVING	ITEM GROUP	ITEM
GIVING-(THROUGH)	ITEM GROUP	ITEM
GIVING-THROUGH	ITEM GROUP	ITEM
GIVING-IN	GROUP	GROUP
PROCESSES-IDMS	IDMS-SUBSCHEMA	IDMS-SUBSCHEMA
PROCESSES-IMS-CONTAINS	STRUCTURE-PCB GSAM-PCB OUTPUT-MESSAGE-PCB	STRUCTURE-PCB
PROCESSES-SUDS	SUDS-SUBSCHEMA	SUDS-SUBSCHEMA
QUALIFIED-ON	ITEM GROUP DLI -SEQU-KEY-PHYSICAL-SEG DLI -SEQU-KEY-INDEX-PTR-SEG INDEX-SEARCH-FIELD SYSTEM-RELATED-FIELD/CK SYSTEM -RELATED-FIELD/SX CONCATENATED-KEY	ITEM
SEE	Any member type	SYSTEM
SELECTING	ITEM GROUP	ITEM
SOURCE-SSR	SYSTEM-2000-SCHEMA-RECORD GROUP S2K-SUBSCHEMA-RECORD	SYSTEM-2000-SCHEMA-RECORD

Clause	Permitted member types	Dummy
SSAS	PHYSICAL-SEGMENT UNIDIRECTIONAL-LOG-CHILD PHYSICALLY-PAIRED-LOG-CHILD REAL-PAIRED-LOG-CHILD VIRTUAL-PAIRED-LOG-CHILD DESTINATION-PARENT-SEG SOURCE-SEGMENT DEST-PARENT/SOURCE SEGMENT TARGET-SEGMENT DEST-PARENT/TARGET-SEGMENT SOURCE/TARGET-SEGMENT DEST-PARENT/SOURCE/ TARGET-SEG LOGICAL-SEGMENT LOGICAL-CONCATENATED-SEG INDEX-POINTER-SEGMENT	PHYSICAL-SEGMENT
SUBSCHEMA	S2K-SUBSCHEMA-RECORD	S2K-SUBSCHEMA-RECORD
TARGET-SSR	SYSTEM-2000-SCHEMA-RECORD GROUP S2K-SUBSCHEMA-RECORD	SYSTEM-2000-SCHEMA-RECORD
USER-PASSWORD	ITEM	ITEM
VIEWS	SYSTEM-2000-SCHEMA-RECORD	SYSTEM-2000-SCHEMA-RECORD

Relationships Permitted from SYSTEM-2000-DATABASE Member Types

This table lists:

- The clauses in the SYSTEM-2000-DATABASE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	**Permitted member types**	**Dummy**
ACCESSES	ITEM GROUP	ITEM
COMPONENT-NUMBER	SYSTEM-2000-SCHEMA-REC GROUP ITEM	SYSTEM-2000-SCHEMA- RECORD
CONTAINS	SYSTEM-2000-SCHEMA-REC GROUP	SYSTEM-2000-SCHEMA- RECORD
FUNCTION-USES	ITEM GROUP	ITEM
MASTER-PASSWORD	ITEM	ITEM
PARENT	SYSTEM-2000-SCHEMA-REC GROUP	SYSTEM-2000-SCHEMA-RECORD
SEE	Any member type	SYSTEM-2000-DATABASE
USER-PASSWORD	ITEM	ITEM

Relationships Permitted from SYSTEM-200-SCHEMA-RECORD Member Types

This table lists:

- The clauses in the SYSTEM-2000-SCHEMA-RECORD member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	**Permitted member types**	**Dummy**
CONTAINS	ITEM GROUP	ITEM
KEYS	ITEM GROUP	ITEM
SEE	Any member type	SYSTEM-2000-SCHEMA-RECORD
USER-EXIT	MODULE	MODULE

Relationships Permitted from SYSTEM-2000-SUBSCHEMA-RECORD Member

This table lists:

- The clauses in the SYSTEM-2000-SUBSCHEMA-RECORD member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	**Permitted member types**	**Dummy**
SEE	Any member type	SYSTEM-2000-SCHEMA-RECORD
SELECTING	ITEM GROUP	ITEM
VIEWS	SYSTEM-2000-SCHEMA-RECORD GROUP	SYSTEM-2000-SCHEMA-RECORD

Relationships Permitted From TOTAL-DATABASE Member Types

This table lists:

- The clauses in the TOTAL-DATABASE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
CONTAINS	TOTAL-MASTER-FILE TOTAL-VARIABLE-FILE	TOTAL-MASTER-FILE
SEE	Any member type	TOTAL DATABASE

Relationships Permitted From TOTAL-MASTER-FILE Member Types

This table lists:

- The clauses in the TOTAL-MASTER-FILE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
CONTAINS	ITEM GROUP	ITEM
CONTROL	ITEM GROUP	ITEM
LINKED-TO	TOTAL-VARIABLE-FILE	TOTAL-VARIABLE-FILE
SEE	Any member type	TOTAL-MASTER-FILE

Relationships Permitted From TOTAL-VARIABLE-FILE Member Types

This table lists:

- The clauses in the TOTAL-VARIABLE-FILE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
CONTAINS	ITEM GROUP	ITEM
IF	ITEM	ITEM
LINK	ITEM GROUP	ITEM
LINK-TO	TOTAL-MASTER-FILE	TOTAL-MASTER-FILE
SEE	Any member type	TOTAL-VARIABLE-FILE

Relationships Permitted from ENTITY Member Types

This table lists:

- The clauses in the ENTITY member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member-type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
IDENTIFIER	ITEM GROUP	ITEM
MULTI-ATTRIBUTES	ITEM GROUP	ITEM
ONE-ATTRIBUTES	ITEM GROUP	ITEM
SUB-ENTITIES	ENTITY	ENTITY
ONE-ASSOCIATIONS	ENTITY	ENTITY
MULTI-ASSOCIATIONS	ENTITY	ENTITY
SEE	Any member type	ENTITY

Relationships Permitted from FORMAT Member Types (DesignManager)

When defining a member type based on a FORMAT member type in a DesignManager environment, the syntax is as defined under ControlManager.

Refer to "Relationships Permitted from FORMAT Member Types" on page 71.

Relationships Permitted From GROUP Member Types (DesignManager)

When defining a member type based on a GROUP member type in a DesignManager environment, the full syntax as defined under DataManager is available.

Refer to page "Relationships Permitted from GROUP Member Types" on page 78.

Relationships Permitted From ITEM Member Types (DesignManager)

When defining a member type based on an ITEM member type in a DesignManager environment, the full syntax as defined under DataManager is available.

Refer to page "Relationships Permitted from ITEM Member Types" on page 85.

Relationships Permitted From USERVIEW Member Types

This table lists:

- The clauses in the USERVIEW member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
DOMAIN	ITEM GROUP	ITEM
LEFT-HAND-SIDE	ITEM GROUP	ITEM

Clause	Permitted member types	Dummy
RIGHT-HAND-SIDE	ITEM GROUP	ITEM
SEE	Any member type	USERVIEW
SUB-DOMAIN	ITEM GROUP	ITEM

Relationships Permitted from VIEWSET Member Types

This table lists:

- The clauses in the VIEWSET member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
CONTAINS	VIEWSET ENTITY USERVIEW	VIEWSET
SEE	Any member type	VIEWSET

Relationships Permitted from TRANSLATION-RULE Member Types

This table lists:

- The clauses in the TRANSLATION-RULE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
SEE	Any member type	TRANSLATION-RULE

Relationships Permitted From DL1-AREA Member Types

This table lists:

- The clauses in the DL1-AREA member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
CONTAINS	PHYSICAL-SEGMENT UNIDIRECTIONAL-LOG-CHILD PHYSICALLY-PAIRED-LOG-CHILD REAL-PAIRED-LOG-CHILD VIRTUAL-PAIRED-LOG-CHILD DESTINATION-PARENT-SEG SOURCE-SEGMENT DEST-PARENT/SOURCE-SEGMENT TARGET-SEGMENT DEST-PARENT/TARGET-SEGMENTSS OURCE/TARGET-SEGMENT DEST-PARENT/SOURCE/TARGET-SEG LOGICAL-SEGMENT LOGICAL-CONCATENATED-SEG INDEX-POINTER-SEGMENT	PHYSICAL-SEGMENT
ALL-SEGMENTS	STRUCTURE-PCB GSAM-PCB OUTPUT-MESSAGE-PCB	STRUCTURE-PCB
SEE	Any member type	DL1-AREA

Relationships Permitted from REUSABLE-CODE Member Types

This table lists:

- The clauses in the REUSABLE-CODE member type that may refer to other member types in a hierarchy.
- The member types to which reference is permitted. The member type listings in this table are also the permitted values of the member type in the IS or BASED-ON clause of the MEMBER-TYPE members.
- The type of dummy that is created if the clause refers to a member that does not exist.

Except for the column headed Dummy, a reference in the table to a base member type should be taken to include any member type modeled on that base member type.

Clause	Permitted member types	Dummy
INPUTS	Any DATA member type	ITEM
OUTPUTS	Any DATA member type	ITEM
UPDATES	Any DATA member type	ITEM

Clause	Permitted member types	Dummy
PARAMETERS	Any DATA member type	ITEM
PASSING	Any DATA member type	ITEM
CONTAINS	Any DATA member type	ITEM
CALLS	REUSABLE-CODE	REUSABLE-CODE
SEE	Any member type	REUSABLE-CODE

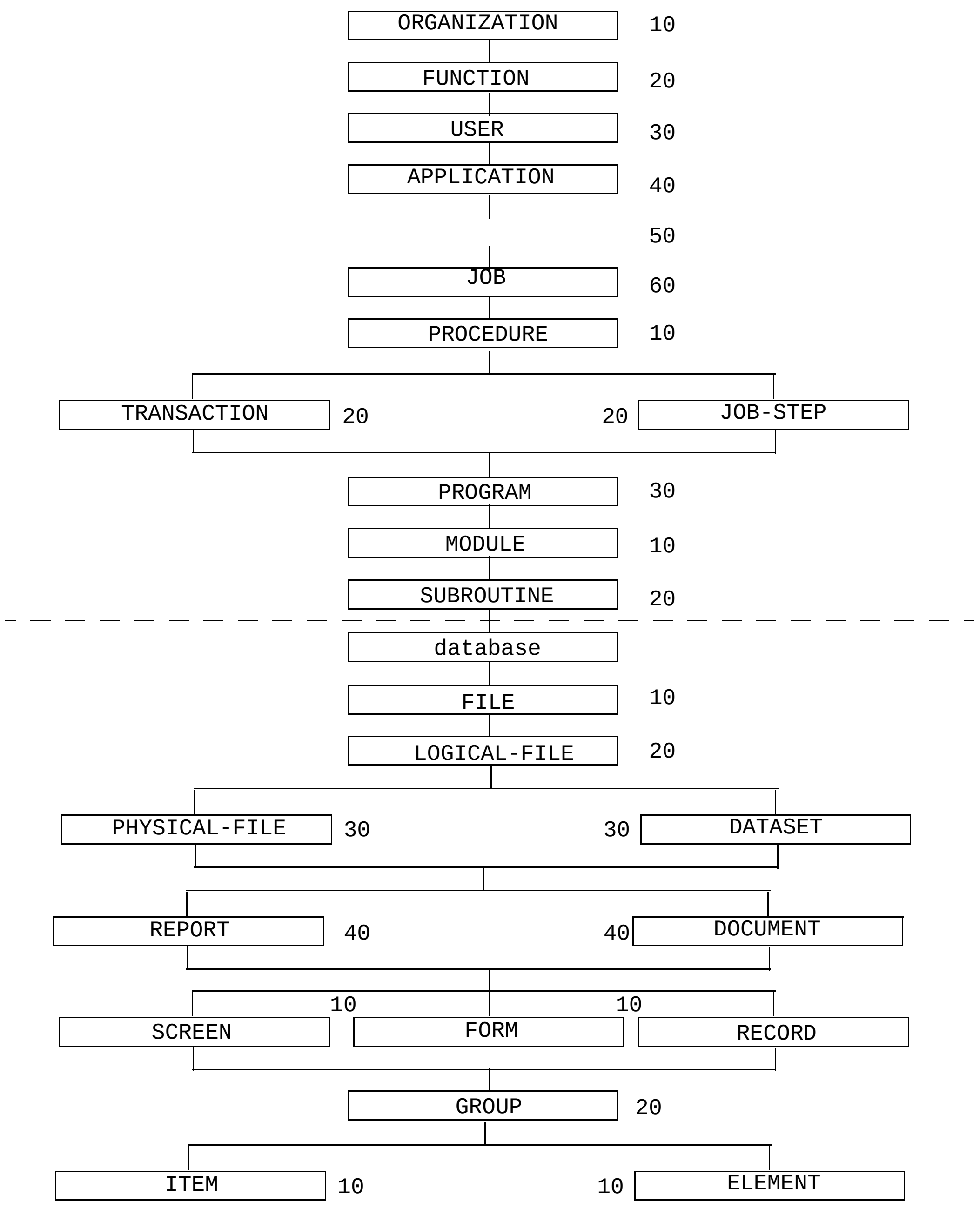
Appendix B

Language Versions of the Extended Data Processing Structure

This chapter contains diagrams illustrating EDPS structure in each of these languages.

Language	**Page**
Danish	114
Dutch	116
English	112
Finnish	118
French	120
German	122
Italian	124
Norwegian	126
Spanish	128
Swedish	130

Figure 5 • EDPS Structure - English



English -

ORGANIZATION

FUNCTION

USER

APPLICATION

$\boxed{\text{SYSTEM}}$

JOB

PROCEDURE

TRANSACTION

JOB-STEP

$\boxed{\text{PROGRAM}}$

$\boxed{\text{MODULE}}$

SUBROUTINE

- -

database

$\boxed{\text{FILE}}$

LOGICAL-FILE

PHYSICAL-FILE

DATASET

REPORT

DOCUMENT

SCREEN

FORM

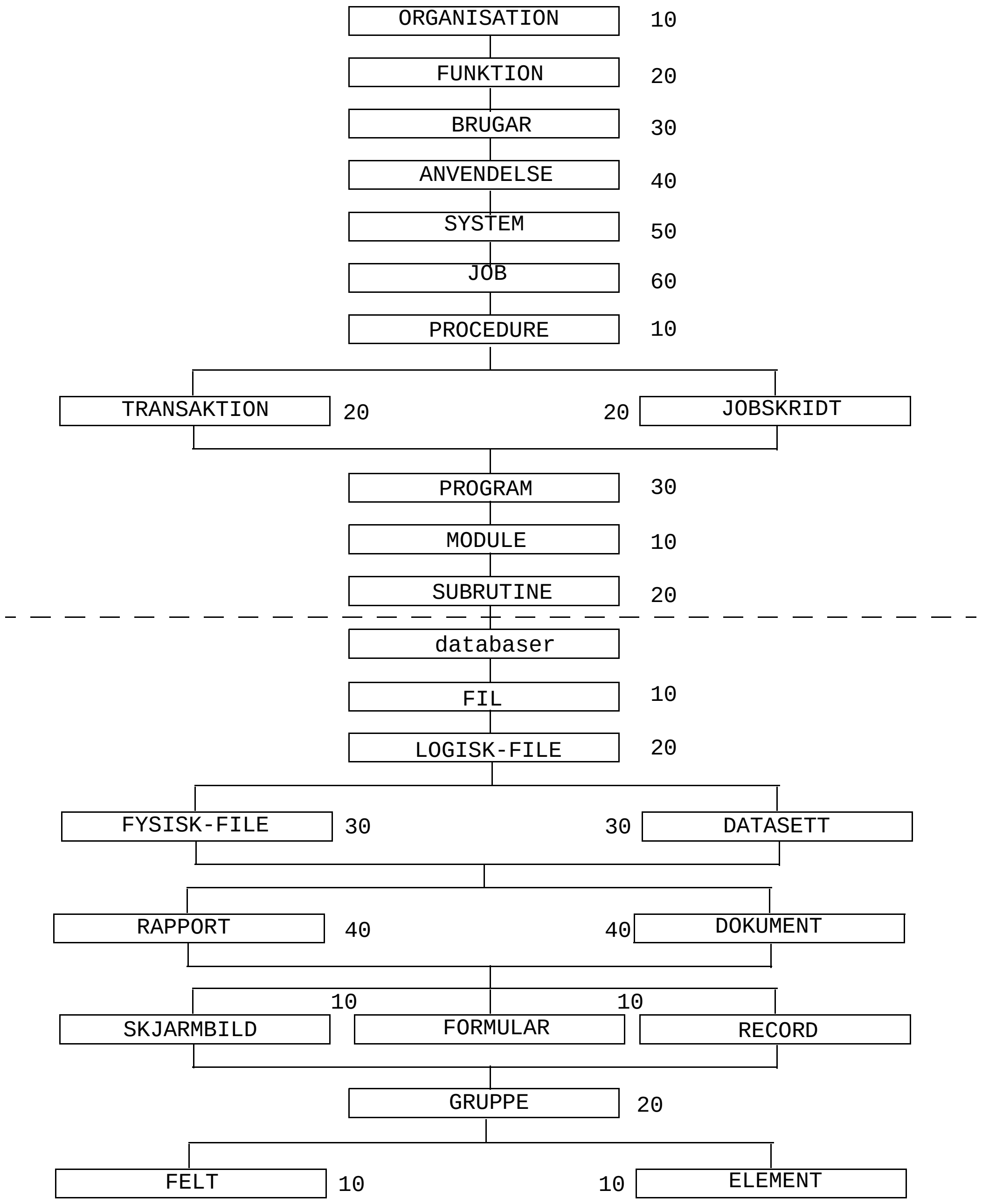
RECORD

$\boxed{\text{GROUP}}$

$\boxed{\text{ITEM}}$

ELEMENT

Figure 6 • EDPS Structure - Danish



Danish -

ORGANISATION

FUNKTION

BRUGAR

ANVENDELSE

$\boxed{\text{SYSTEM}}$

JOB

PROCEDURE

TRANSAKTION

JOBSKRIDT

$\boxed{\text{PROGRAM}}$

$\boxed{\text{MODULE}}$

SUBRUTINE

- -

databaser

$\boxed{\text{FIL}}$

LOGISK-FIL

FYSISK-FIL

DATASETT

RAPPORT

DOKUMENT

SKJARMBILD

FORMULAR

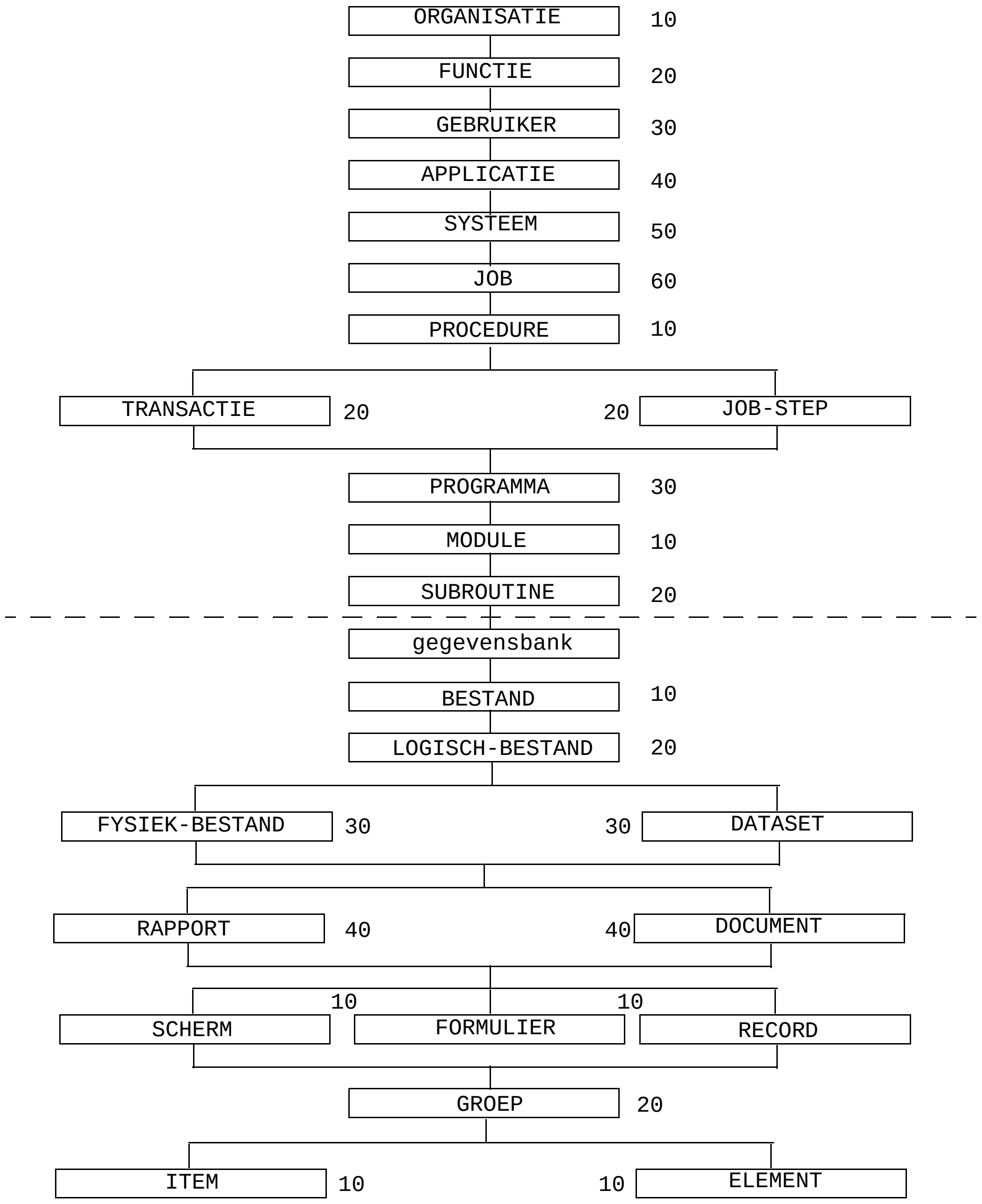
RECORD

$\boxed{\text{GRUPPE}}$

$\boxed{\text{FELT}}$

ELEMENT

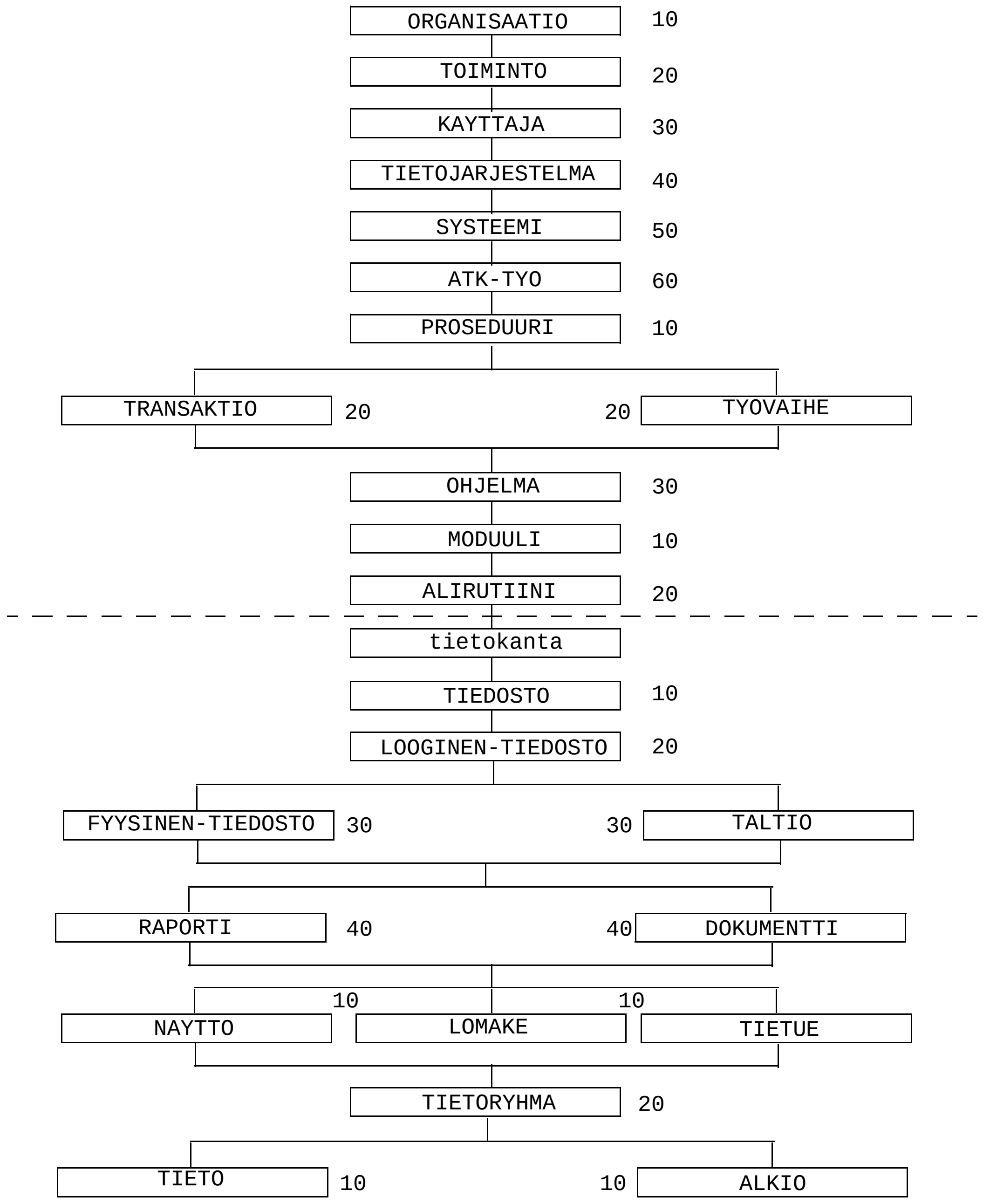
Figure 7 • EDPS Structure - Dutch



Dutch - continued

```
ORGANISATIE

FUNCTIE

GEBRUIKER

APPLICATIE

SYSTEEM

JOB

PROCEDURE

TRANSACTIE

JOB-STEP

PROGRAMMA

MODULE

SUBROUTINE
- - - - - - - - - - - - - - - - - - - - - - - - - - - - -
gegevensbank

BESTAND

LOGISCH-BESTAND

FYSIEK-BESTAND

DATASET

RAPPORT

DOCUMENT

SCHERM

FORMULIER

RECORD

GROEP

ITEM

ELEMENT
```

Figure 8 • EDPS Structure - Finnish



Finnish -

ORGANIZAATIO

TOIMINTO

KAYTTAJA

TIETOJARJESTELMA

$\boxed{\text{SYSTEEMI}}$

ATK-TYO

PROSEDUURI

TRANSAKTIO

TYOVAIHE

$\boxed{\text{OHJELMA}}$

$\boxed{\text{MODUULI}}$

ALIRUTIINI

---

tietokanta

$\boxed{\text{TIEDOSTO}}$

LOOGINEN-TIEDOSTO

FYYSINEN-TIEDOSTO

TALTIO

RAPORTTI

DOKUMENTTI

NAYTTO

LOMAKE

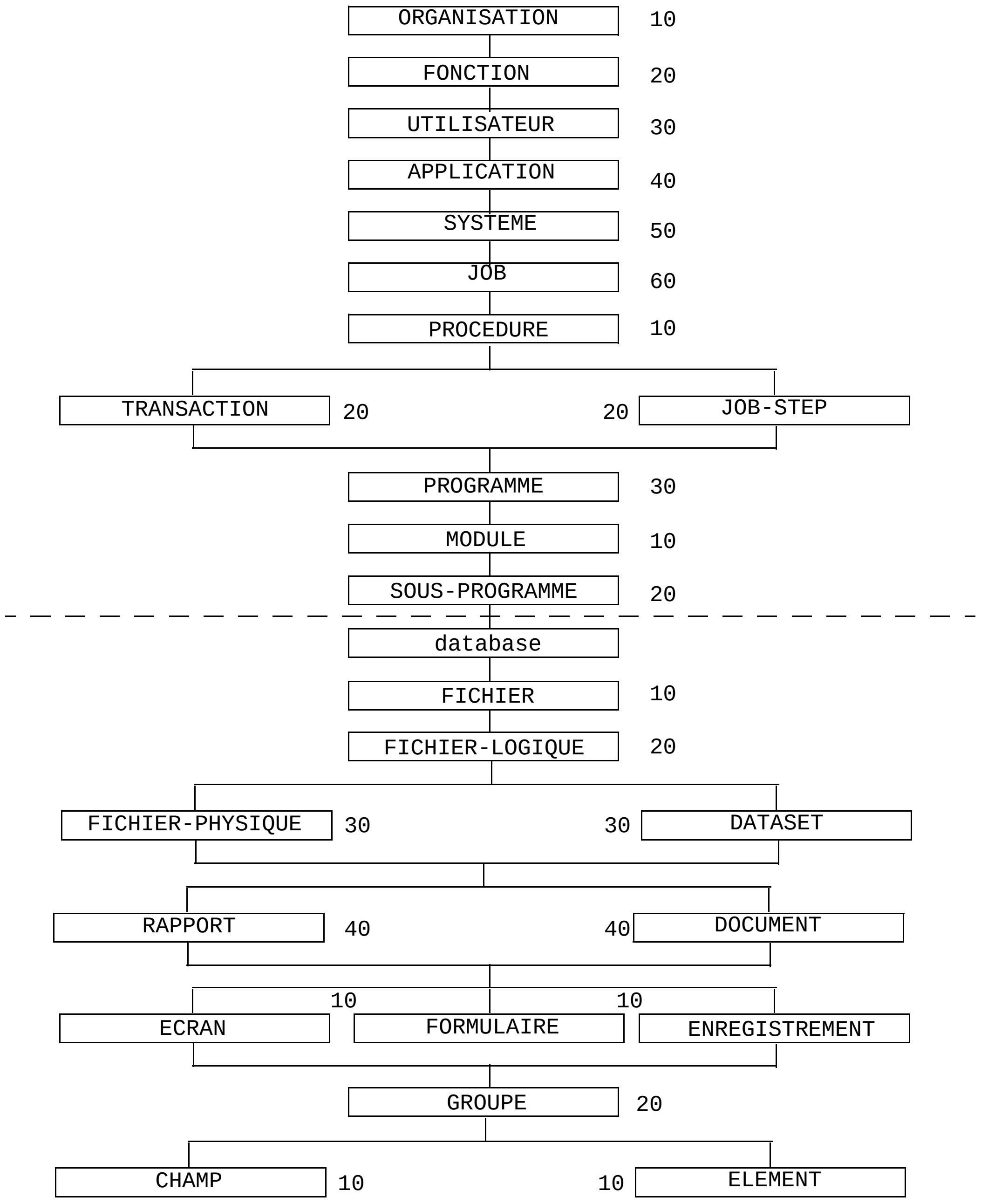
TIETUE

$\boxed{\text{TIETORYHMA}}$

$\boxed{\text{TIETO}}$

ALKIO

Figure 9 • EDPS Structure - French



French -

ORGANISATION

FONCTION

UTILISATEUR

APPLICATION

$\boxed{\text{SYSTEME}}$

JOB

PROCEDURE

TRANSACTION

JOB-STEP

$\boxed{\text{PROGRAMME}}$

$\boxed{\text{MODULE}}$

SOUS-PROGRAMME

- -

database

$\boxed{\text{FICHIER}}$

FICHIER-LOGIQUE

FICHIER-PHYSIQUE

DATASET

RAPPORT

DOCUMENT

ECRAN

FORMULAIRE

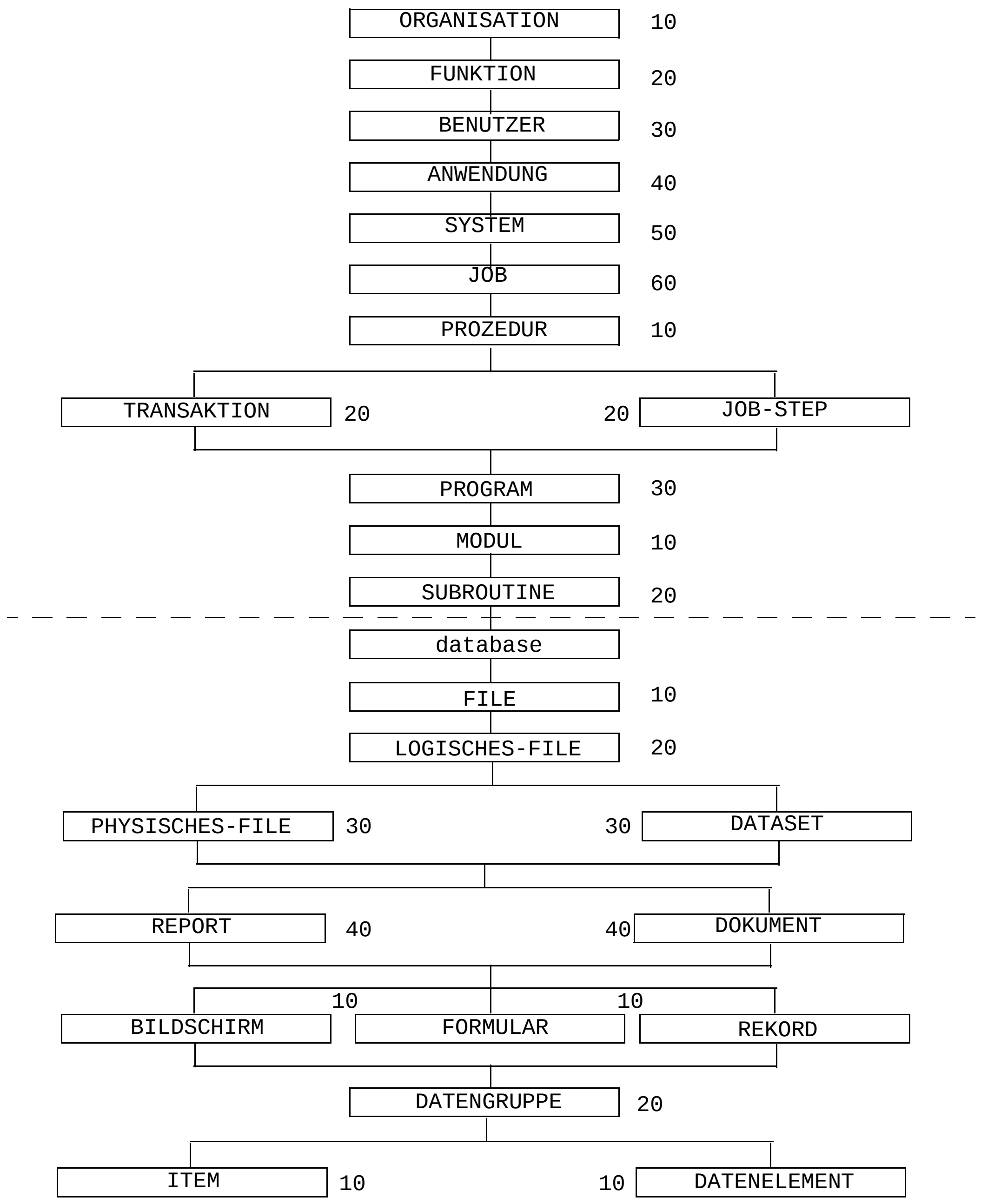
ENREGISTREMENT

$\boxed{\text{GROUPE}}$

$\boxed{\text{CHAMP}}$

ELEMENT

Figure 10 • EDPS Structure - German



German -

ORGANISATION

FUNKTION

BENUTZER

ANWENDUNG

$\boxed{\text{SYSTEM}}$

JOB

PROZEDUR

TRANSAKTION

JOB-STEP

$\boxed{\text{PROGRAM}}$

$\boxed{\text{MODUL}}$

SUBROUTINE

- -

database

$\boxed{\text{FILE}}$

LOGISCHES-FILE

PHYSISCHES-FILE

DATASET

REPORT

DOKUMENT

BILDSCHIRM

FORMULAR

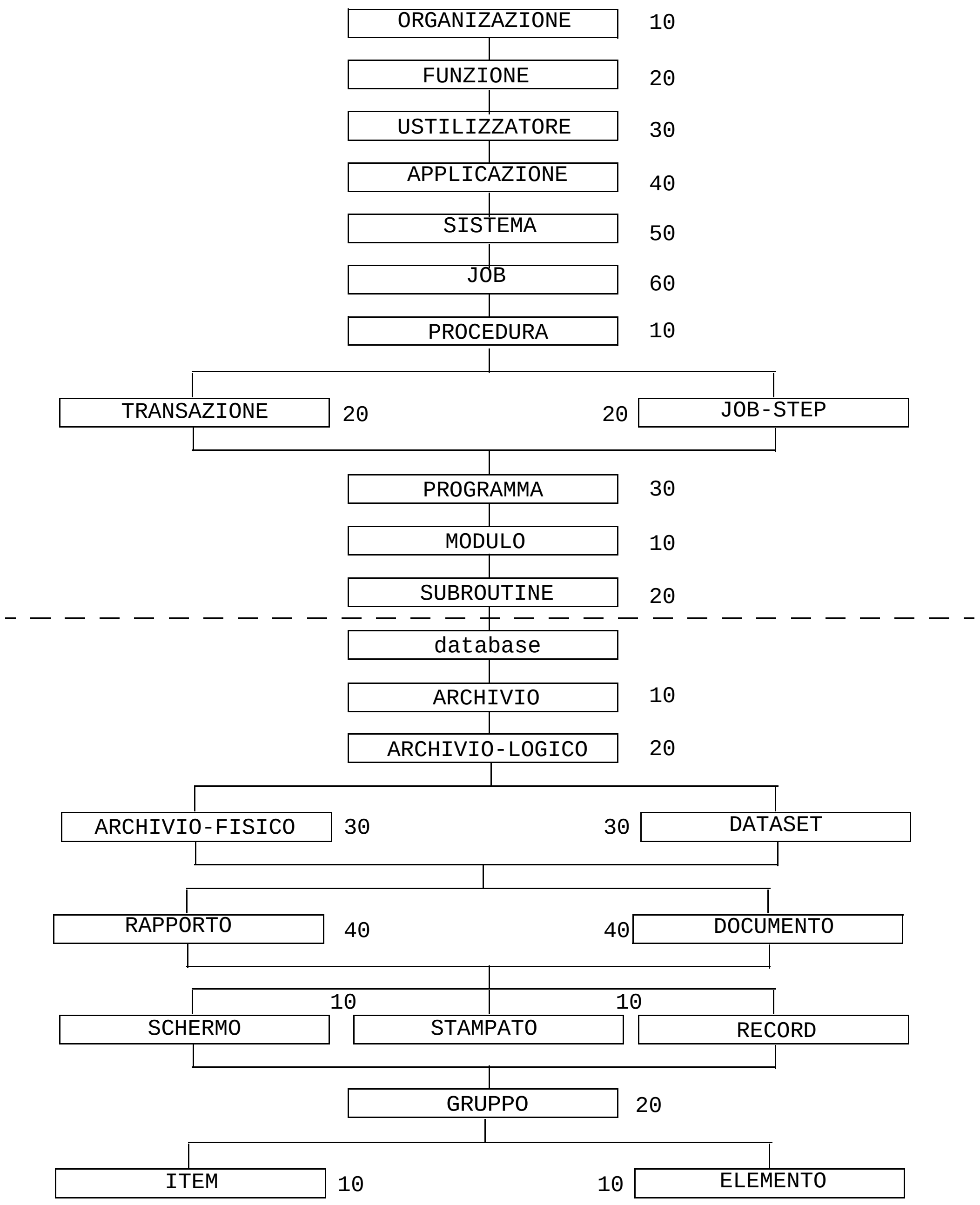
REKORD

$\boxed{\text{DATENGRUPPE}}$

$\boxed{\text{ITEM}}$

DATAENELEMENT

Figure 11 • EDPS Structure - Italian



Italian - continued

ORGANIZZAZIONE

FUNZIONE

UTILIZZATORE

APPLICAZIONE

$\boxed{\text{SISTEMA}}$

JOB

PROCEDURA

TRANSAZIONE

JOB-STEP

$\boxed{\text{PROGRAMMA}}$

$\boxed{\text{MODULO}}$

SUBROUTINE

---

database

$\boxed{\text{ARCHIVIO}}$

ARCHIVIO-LOGICO

ARCHIVIO-FISICO

DATASET

RAPPORTO

DOCUMENTO

SCHERMO

STAMPATO

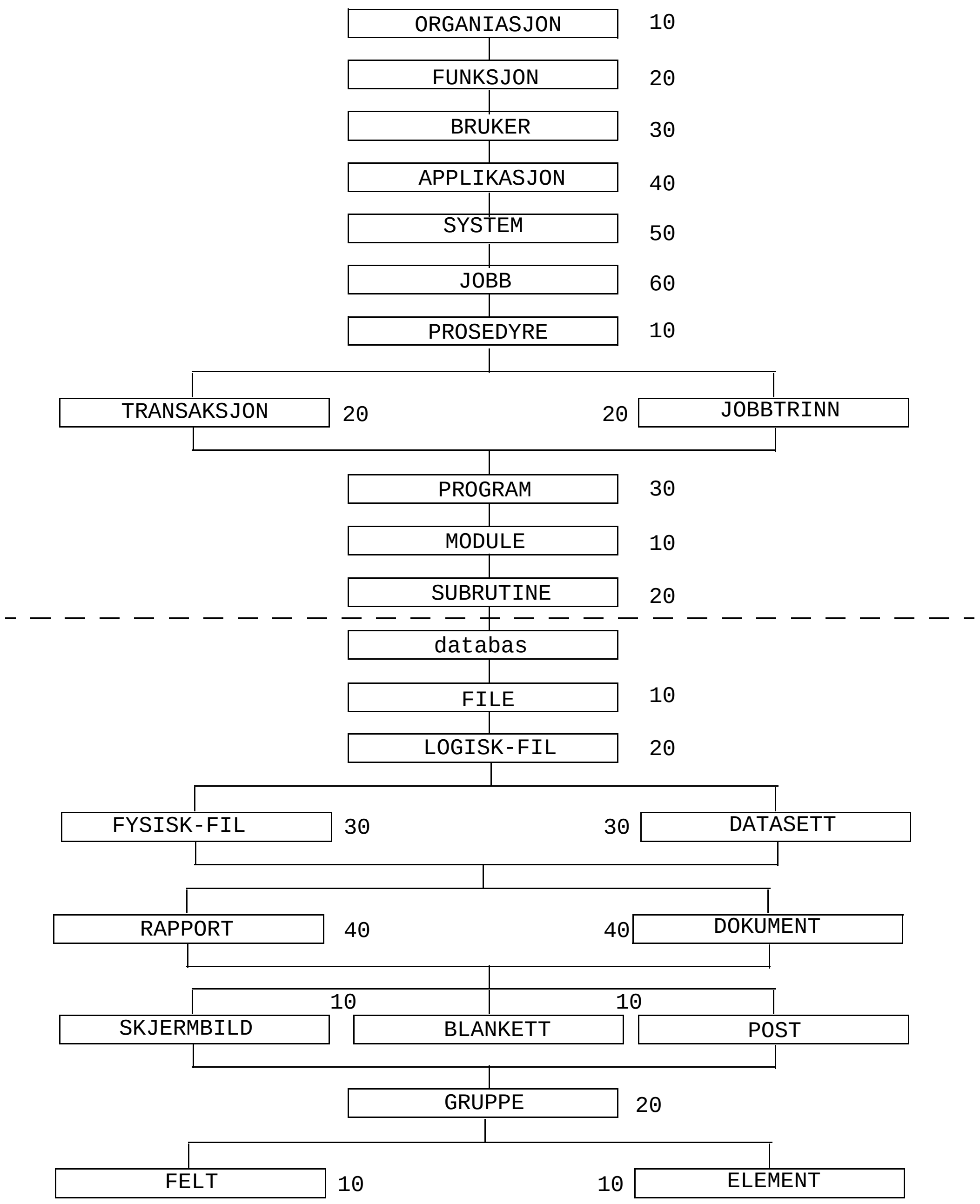
RECORD

$\boxed{\text{GRUPPO}}$

$\boxed{\text{ITEM}}$

ELEMENTO

Figure 12 • EDPS Structure - Norwegian



Norwegian -

ORGANIASJON

FUNKSJON

BRUKE

APPLIKASJON

$\boxed{\text{SYSTEM}}$

JOBB

PROSEDYRE

TRANSAKSJON

JOBBTRINN

$\boxed{\text{PROGRAM}}$

$\boxed{\text{MODUL}}$

SUBRUTIN

---

databas

$\boxed{\text{LOGISK-FIL}}$

FYSISK-FIL

DATASETT

RAPPORT

DOKUMENT

SKJERMBILD

BLANKETT

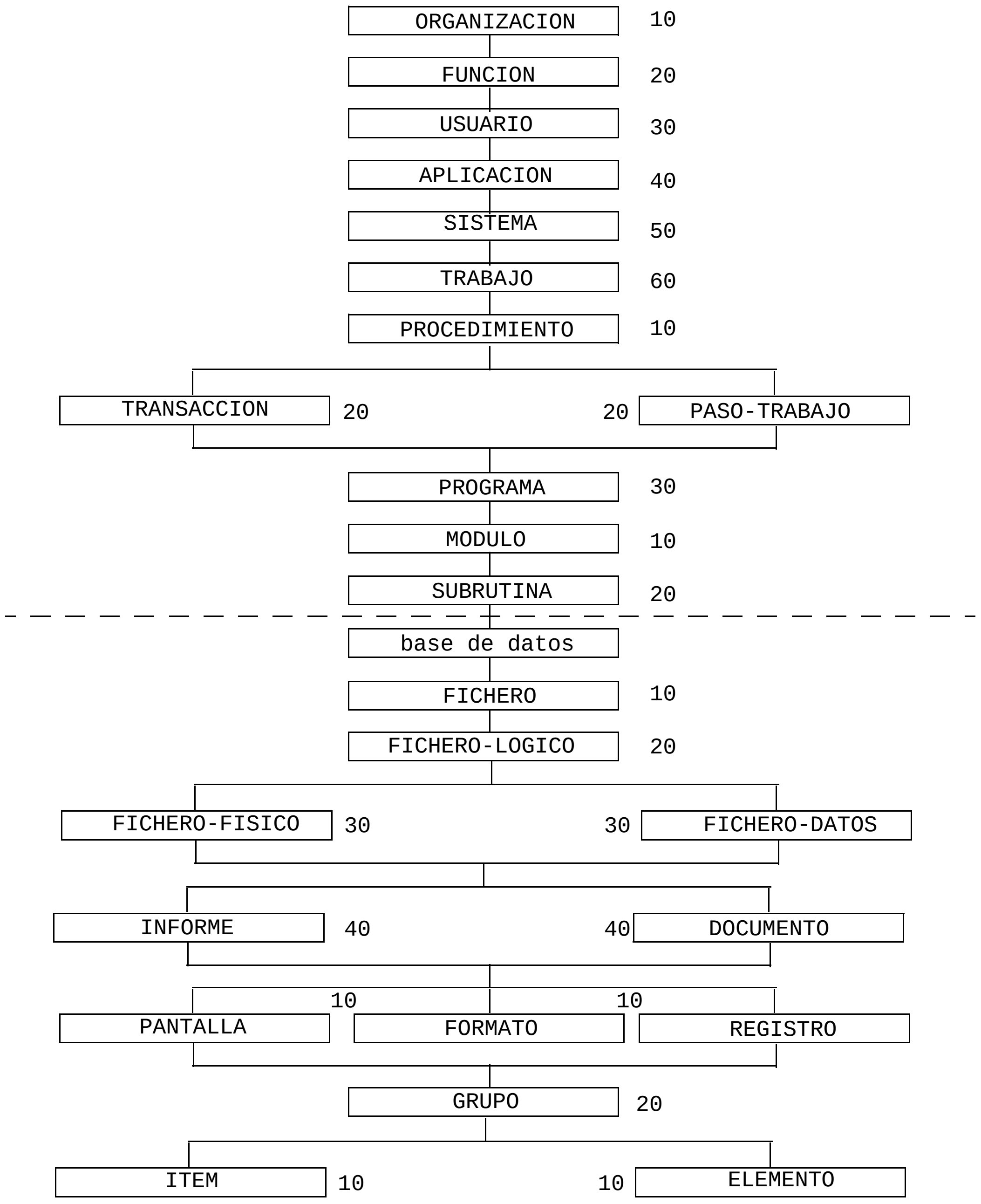
POST

$\boxed{\text{GRUPPE}}$

$\boxed{\text{FELT}}$

ELEMENT

Figure 13 • EDPS Structure - Spanish



Spanish -

ORGANIZACION

FUNCION

USUARIO

APLICACION

$\boxed{\text{SISTEMA}}$

TRABAJO

PROCEDIMIENTO

TRANSACCION

PASO-TRABAJO

$\boxed{\text{PROGRAMA}}$

$\boxed{\text{MODULO}}$

SUBRUTINA

---

base de datos

$\boxed{\text{FICHERO}}$

FICHERO-LOGICO

FICHERO-FISICO

FICHERO-DATOS

INFORME

DOCUMENTO

PANTALLA

FORMATO

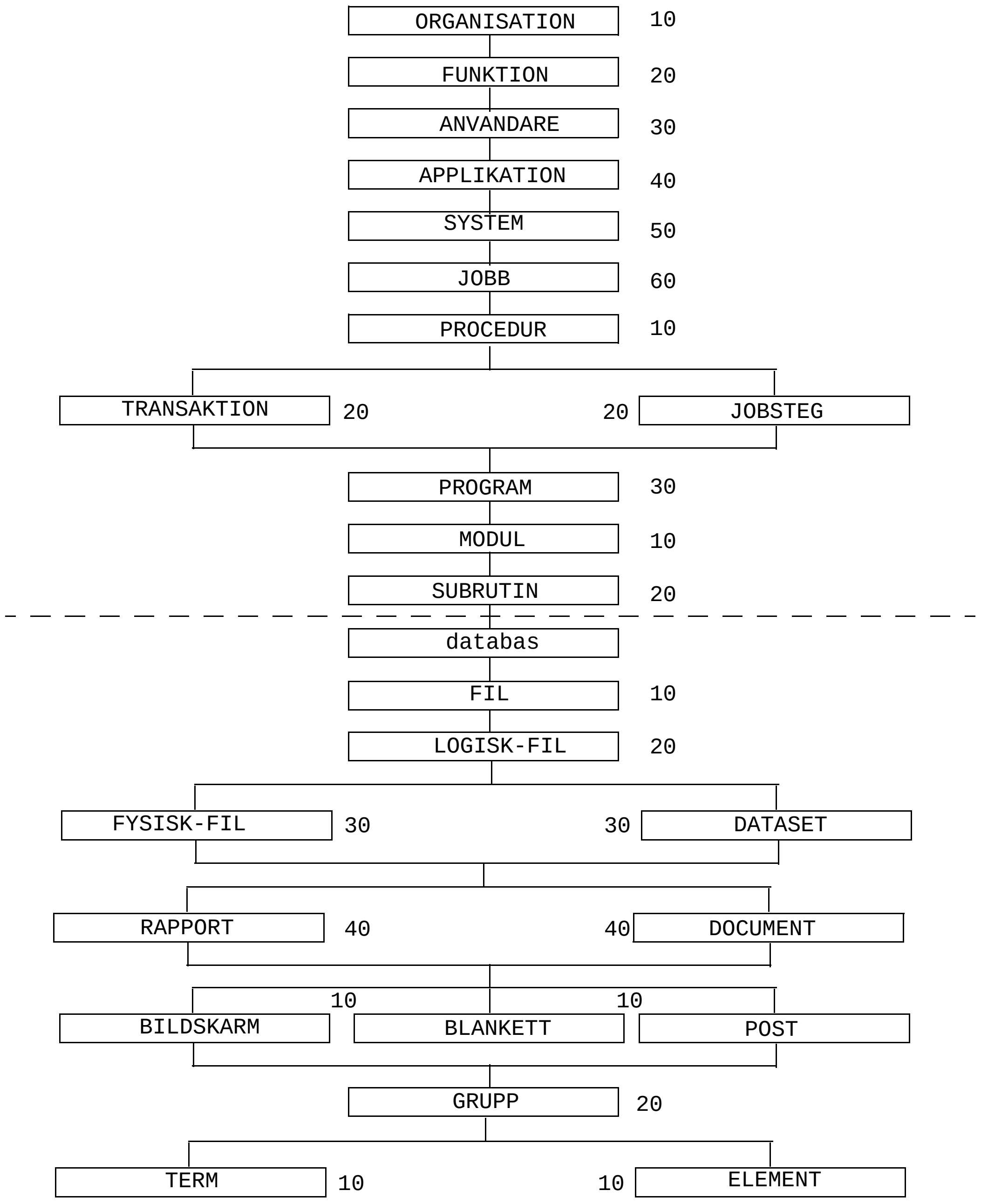
REGISTRO

$\boxed{\text{GRUPO}}$

$\boxed{\text{ITEM}}$

ELEMENTO

Figure 14 • EDPS Structure - Swedish



Swedish - continued

ORGANISATION

FUNKTION

ANVANDARE

APPLIKATION

$\boxed{\text{SYSTEM}}$

JOBB

PROCEDUR

TRANSAKTION

JOBSTEG

$\boxed{\text{PROGRAM}}$

$\boxed{\text{MODUL}}$

SUBRUTIN

---

database

$\boxed{\text{FIL}}$

LOGISK-FIL

FYSISK-FIL

DATASET

RAPPORT

DOKUMENT

BILDSKARM

BLANKETT

POST

$\boxed{\text{GRUPP}}$

$\boxed{\text{TERM}}$

ELEMENT

Appendix C

Relationships Permitted from IMS Member Types

DataManager IMS second-level member types are listed below. Relationships that are to be permitted from IMS member types must be specified in the MEMBER-TYPE members at this level. The clause keywords that are permitted in the RELATIONSHIPS VIA clause of the relevant MEMBER-TYPE member and the member types to which reference is permitted will be provided by an Amendment List.

Databases

- DL1-GSAM-DATABASE
- DL1-SIMPLE-HSAM-DATABASE
- DL1-HSAM-DATABASE
- DL1-SIMPLE-HISAM-DATABASE
- DL1-HISAM-DATABASE
- DL1-HIDAM-DATABASE
- DL1-HDAM-DATABASE
- DL1-SECONDARY-INDEX-DATABASE
- DL1-SHARES-WITH-SECNDRY-INDEX-DB
- DL1-PRIMARY-INDEX-DATABASE
- DL1-LOGICAL-DATABASE

Segments

- PHYSICAL-SEGMENT
- UNIDIRECTIONAL-LOGICAL-CHILD
- PHYSICALLY-PAIRED-LOGICAL-CHILD
- REAL-PAIRED-LOGICAL-CHILD
- VIRTUAL-PAIRED-LOGICAL-CHILD
- DESTINATION-PARENT-SEGMENT
- SOURCE-SEGMENT
- DESTINATION-PARENT/SOURCE-SEGMENT
- TARGET-SEGMENT
- DEST-PARENT/TARGET-SEG
- SOURCE/TARGET-SEG
- DEST-PARENT/SOURCE/TARGET-SEG
- LOGICAL-SEGMENT
- LOGICAL-CONCATENATED-SEGMENT
- INDEX-POINTER-SEGMENT
- PRIMARY-INDEX-POINTER-SEGMENT

PCBs

- DL1-STRUCTURE-PCB
- DL1-GSAM-PCB
- DL1-OUTPUT-MESSAGE-PCB

Index